NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

DISSERTATION

CROSS-DOMAIN NETWORK FAULT LOCALIZATION

by

William D. Fischer

June 2009

Dissertation Supervisor: Geoffrey G. Xie

Approved for public release; distribution is unlimited.

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 2009	Dissertation

4. TITLE AND SUBTITLE
Cross-Domain Network Fault Localization

5. FUNDING NUMBERS

6. AUTHORS William D. Fischer

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT(*maximum 200 words*)

Prior research has focused on intra-domain fault localization leaving the cross-domain problem largely unaddressed. Faults often have widespread effects, which if correlated, could significantly improve fault localization. For both competitive and security reasons, domain managers hesitate to share fault observations even when doing so may significantly ease fault localization. This dissertation presents a characterization of the problem space in terms of inference accuracy, privacy, and scalability, and provides a framework to evaluate any design in the design spectrum. This framework not only explicitly models the inference accuracy and privacy requirements for discussing and reasoning over cross-domain problems, but also addresses scalability impacts and facilitates the re-use of existing fault localization algorithms while enforcing domain privacy policies. The dissertation provides a graph-digest-based approach with which participating network domains can exchange abstracted graphs that represent network fault propagation models. The research explores feasibility of this approach via implementation of an inference graph-based design in a cross-domain network setting. The results show a substantial improvement in cross-domain fault localization accuracy and inference speed by using the inference-graph-digest based approach.

14. SUBJECT TERMS
Networking, Fault Localization, Cross-Domain, Bayesian

15. NUMBER OF PAGES 135

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UU

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for public release; distribution is unlimited.

CROSS-DOMAIN NETWORK FAULT LOCALIZATION

William D. Fischer
Lieutenant Colonel, United States Army
B.S., College of William and Mary, 1989
M.S., Naval Postgraduate School, 2001
Submitted in partial fulfillment of the
requirements for the degree of

DOCTOR OF PHILOSOPHY IN COMPUTER SCIENCE
from the
NAVAL POSTGRADUATE SCHOOL
June 2009

Author: _____
William D. Fischer

Approved by: _____
Geoffrey G. Xie
Professor of Computer Science, Dissertation Supervisor

Craig H. Martell Associate Professor of Computer Science	Mikhail Auguston Associate Professor of Computer Science
Joel D. Young Lieutenant Colonel, United States Air Force Assistant Professor of Computer Science	Craig W. Rasmussen Associate Professor of Applied Mathematics

Approved by: _____
Peter J. Denning, Chair, Department of Computer Science

Approved by: _____
Doug Moses, Associate Provost for Academic Affairs

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Prior research has focused on intra-domain fault localization leaving the cross-domain problem largely unaddressed. Faults often have widespread effects, which if correlated, could significantly improve fault localization. For both competitive and security reasons, domain managers hesitate to share fault observations even when doing so may significantly ease fault localization. This dissertation presents a characterization of the problem space in terms of inference accuracy, privacy, and scalability, and provides a framework to evaluate any design in the design spectrum. This framework not only explicitly models the inference accuracy and privacy requirements for discussing and reasoning over cross-domain problems, but also addresses scalability impacts and facilitates the re-use of existing fault localization algorithms while enforcing domain privacy policies. The dissertation provides a graph-digest-based approach with which participating network domains can exchange abstracted graphs that represent network fault propagation models. The research explores feasibility of this approach via implementation of an inference graph-based design in a cross-domain network setting. The results show a substantial improvement in cross-domain fault localization accuracy and inference speed by using the inference-graph-digest based approach.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I especially thank my dissertation supervisor, Dr. Geoffrey Xie, for investing his time and energy to mentor me from day one. His constant encouragement and passion for science helped to elevate my level of thinking and to shape me as an academic. His expertise and experience were critical to this research endeavor.

I thank Lieutenant Colonel Joel Young, Ph.D., for his patience, guidance, and friendship. I am eternally grateful for his insight, which was always on the mark, and for the many hours he invested advising me during the research.

I thank my other committee members Dr. Craig Martell, Dr. Mikhail Auguston, and Dr. Rasmussen for their support throughout my research. Their assistance, feedback, and guidance along the way proved invaluable.

I enjoyed sharing my experience with a great group of doctoral students. Having a mutual-support network and peers to bounce ideas off of made the process much more enjoyable.

Most importantly, I want to thank my family for their support throughout this endeavor. I could not have reached this milestone without Natalie's unwavering love and understanding. To Andrew and Isabella, you remind me every day of what's most important. Nubia's help along the way with the kids was a tremendous help, and gave me the flexibility I needed when I needed it most.

This research was partially sponsored by the NSF under grants CNS-0520210 and CNS-0721574. Views and conclusions contained in this research are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NSF, or the U.S. government.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

Computer network faults happen frequently and finding the source of failure is a non-trivial task. Although much progress has been made locating faults within network domains, finding faults that affect multiple domains, or *cross-domain fault localization*, remains under-researched. Network domain administrators currently perform fault diagnosis in isolation, without benefit from evidence observed in other domains. Today's highly connected networks need collaboration to locate complex failures, but privacy concerns tend to prevent cooperation across network boundaries. This research proposes a cooperative approach in which domain administrators share *some* data to find these elusive faults, while preserving privacy for sensitive network domain properties.

Faults in a network occur often and in complex ways, and it is well-documented that managers must respond to these failures on a regular basis [19, 23, 46]. There are a wide variety of components in a network that can fail and fiber cuts, router mis-configuration, and power and maintenance outages are becoming more common [13, 23, 48]. Diversity of network elements within a network continues to grow [8]. The heterogeneity of elements in a network adds complication to all aspects of managing a network. Dependencies between network elements are not always deterministic, increasing the difficulty of correlating observations about network state. Failure durations can vary, increasing the difficulty of correlating observation data and further complicating diagnosis [19, 48]. Serious faults can be undetected and may not be able to be rapidly localized [23].

When observations of network state arising from a network fault propagate across domain boundaries, the fault is described as *cross-domain*. Troubleshooting faults is a challenging task—it is even more difficult when trying to troubleshoot cross-domain issues without knowledge of fault observations and network structure from neighboring network domains. Acquiring knowledge of the needed observations

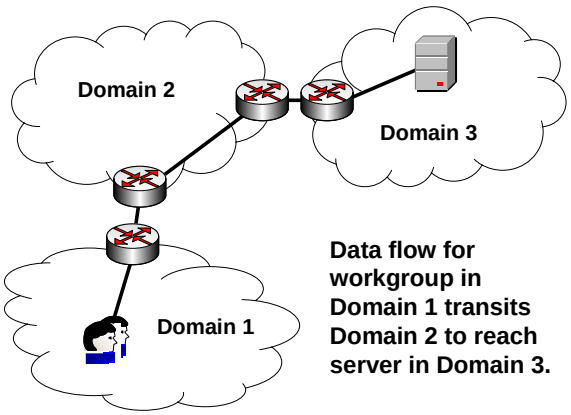


Figure 1.1. Simple Failure Scenario

and network topology is further complicated by the fact that it is risky, for both competitive and security reasons, for domain managers to share this information even when the sharing might ease fault localization. With business processes migrating to web-services, implemented in the "cloud" and built on protocols such as SOAP (Simple Object Access Protocol), the likelihood of network faults impacting multiple domains approaches unity.

Faults often have widespread effects, which if correlated, can significantly increase fault localization accuracy. This research defines *inference gain* to be the increase in inference accuracy achieved by correlating additional evidence. Cross-domain network failures can not always be localized without a coordinated effort between domains. Consider the simple failure scenario depicted in Figure 1.1. A work group in Domain 1 must access data from a server in Domain 3 requiring connectivity through Domain 2. Unfortunately, one of the routers in Domain 2 is misconfigured. Other groups and services can reach the server in Domain 3, but users in Domain 1's work group can not. Furthermore, no equipment failures along the path from the work group (Domain 1) to the server (Domain 3) trigger alarms. This is difficult to troubleshoot without cross-domain collaboration, often resulting in "finger pointing." While the fault remains unabated and potentially unnoticed, there

may be observations external to each domain that could help detect and localize the fault. Overcoming obstacles to cross-domain collaboration can realize inference gain to stamp out otherwise ambiguous network errors.

The cross-domain environment introduces a source of complex potential failures. There are more than 10,000 autonomous systems (ASs) in the Internet today, each applying local policies for route selection [42]. Domains in the Internet today are loosely coupled [13], and since cross-domain flows depend on network elements in more than one domain, no single domain has complete control of all risks to the flows. Operators make manual changes in routing policies without fully understanding the impact to other domains, and business arrangements may restrict traffic flow between Autonomous Systems [14]. Links between domains are common points of congestion, and traffic engineering between domains is often achieved through trial and error [14]. Traffic engineering across domains is significantly more complicated than traffic engineering within a domain [14]. This reality implies that complexity in performing fault localization across domains also increases.

Privacy, *scalability*, and *interoperability* issues hinder efforts to achieve accurate cross-domain fault localization. While prior work has stated the importance of these issues [18, 25, 30, 44], review of the literature did not find a formal definition of requirements addressing them. These same issues prevent network collaboration for other types of inference [27, 47]. Network domain managers are often unwilling or not permitted to share detailed internal network architectures and quality-of-service issues with outside agencies, running face-first into the need to share data to successfully troubleshoot networking issues. Automated techniques for finding faults across a large number of domains face serious computational issues and exact computation using belief networks is NP-hard [19]. Interoperability in network-management and fault-isolation techniques is a perennial problem: Different modeling techniques and tools using different algorithms will be employed in various domains. Conflict of information formats and semantics may arise between domains, with each domain's

model assigning a different value to the same parameter. There is a dramatic need for methods enabling cross-domain fault localization efficiently while minimizing the need to share sensitive proprietary information.

Cross-domain fault isolation efforts are hampered by *privacy* issues. Domain managers may be reluctant to share the details of their network dependency graph data with other domain managers. ISPs do not want statistics on the number of problems observed in their network publicly available, and do not want to share their topology and state information with their competitors [25, 27, 44]. Different network providers have proprietary network fault management systems without open interfaces [18]. Each domain manager will have some information about another network domain, such as shared peering points, public web services, and publicly available company information. The vast majority of components in another network, however, can only be modeled as a cloud. Network providers are likely considered competitors and any benefit attained by collaboration to localize faults must outweigh the cost of revealing internal details to the competition.

A cross-domain fault localization approach may not *scale*. There is no central fault management system for the Internet and combining data to resolve a cross-domain failure scenario with a centralized model may not be realistic. Consider a scenario in which a backbone link failure has impacted many domains. In the worst case much of the Internet may need to be mapped into a fault propagation model. Traditional debugging tools do not scale across administrative domains [30]. A typical tier-1 network (also known as an Internet backbone network [24]) has approximately 1,000 routers, supported by two orders of magnitude more access and core transport network elements [23].

To *interoperate*, a cross-domain approach must overcome the heterogeneity of existing fault localization approaches. Even if domain managers collaborate for the purpose of fault localization, they may use different methods of data representation and interpretation. Different domains may be using different inference algorithms,

different tools, and possibly different data schema. Each domain may employ a fault management system that is fundamentally different from those used by other domains with whom they regularly interact. Inference methods used by fault management systems vary widely, and the data from one may not have meaning in another.

Three general approaches are possible for diagnosing cross-domain problems. The first of these, the status quo, is *isolated inference*. In this approach, each domain tries to locate the fault without sharing data with other domains. The second approach, referred to as *full disclosure*, entails full collaboration and data-sharing between domains. A fault propagation model using this approach is equivalent to a global model of all domains involved. While full-disclosure, in general, is unrealistic because of the privacy factor and for scalability reasons, it is included as a baseline model for studying inference gains achievable from information sharing. The third approach, proposed by this research and termed "cooperative", is to implement a design in the design spectrum that lies somewhere between isolated inference and full disclosure. In this third approach domains exchange limited information, e.g., summaries of fault observations, to perform inference while protecting sensitive information. This research focuses on exploring the feasibility of the third approach.

A. PROBLEM STATEMENT AND MAIN HYPOTHESIS

Problem Statement: Cross-domain fault localization is an under-researched area for which no general approach currently exists. External evidence that can improve inference accuracy about network faults is unavailable to domain inference algorithms. Privacy, scalability, and interoperability issues restrict information exchange about these observations.

Main Hypothesis: It is possible to construct a framework to enable managers of separate network domains to share information and achieve inference gain while quantifying privacy preservation of sensitive information.

B. CONTRIBUTIONS

This research makes the following major contributions to the state-of-art for computer network fault localization:

- This research provides a characterization of the problem space for cross-domain fault localization, and provides explicit metrics to evaluate any approach in terms of the core issues of accuracy, privacy, and scalability.
- This research develops the first concrete solution framework providing a feasible, general approach to address cross-domain fault localization. This framework describes a *graph digest* approach that enables domains that use causal graphs to model fault propagation to exchange summary inference information.
- This research provides a first application of the framework using intra-domain fault localization algorithms to locate faults in a cross-domain setting.
- This research provides a first heuristic to learn a network domain's topology from a bipartite causal graph.

C. ORGANIZATION

The outline for the remainder of this dissertation is as follows:

- Chapter II discusses related work in fault localization, fault localization algorithms, and cross-domain fault localization.
- Chapter III describes the approach to model the problem, including metrics to evaluate an approach.
- Chapter IV provides the evaluation methodology.
- Chapter V presents the evaluation results.
- Chapter VI presents the conclusions for this research, and suggests areas of future work.

II. RELATED WORK

This chapter presents the state-of-the-art for intra-domain and cross-domain fault localization. As this research examines whether existing fault localization algorithms can be applied in a cross-domain context, a selection of recent algorithms are presented. This chapter is organized as follows. First, the chapter discusses general concepts of network fault localization. Second, the principle methods of inference used by recent fault localization approaches are surveyed. In particular, SHRINK and SCORE are highlighted because they are used to to evaluate the model in Chapter V. Third, solutions proposed prior to this effort for cross-domain fault localization are discussed. Fourth, network tomography, a current approach used to collect network status data is discussed. Finally, recent work on graph anonymization, which has direct bearing on privacy preservation is described.

A. NETWORK FAULT LOCALIZATION

Failures can stem from many causes, including hardware, software, and configuration errors. Errors may be introduced at each stage of a network's architectural implementation [23]. Fault diagnostic information is subject to errors due to inaccurate models of network dependencies, missing observations, and spurious [39] observations. Typically human operators perform device configuration, resulting in potentially misconfigured devices.

The sheer number of components in a system increases the frequency of failures, and the complexity in locating a failure. Network flows can cross many components in a network and a failure of any one component on the path can sever end-to-end connectivity. A typical tier-1 network has roughly 1,000 routers from different vendors, having different features and playing different roles [23].

Fault localization is the second step of fault diagnosis, which consists of three

steps: fault detection, fault localization, and testing/analysis [8, 37]. Fault detection usually comes from alarms. Network administrators typically employ a fault localization solution to arrive at likely hypotheses to explain the observations about detected faults. Fault localization, a major task in maintaining network service, is the process of determining the actual faults responsible for observed problems in a system [41]. The data typically available to perform fault localization includes potential fault causes, observations of network state, dependencies between causes and observations, prior probabilities of fault causes, and dependencies between fault causes. Fault localization algorithms return a best explanation, which is a set containing the most likely failed root cause or causes, given the model and available evidence. Ideally, a best explanation precisely matches the ground truth.

Domain managers typically employ a fault management system, instrumented with alarms, that infers the best explanation for observed alarms based on the network dependencies. These dependencies are stored in a shared risk database, representing the network components subject to failure [19]. Many of the alarms are based on Simple Network Management Protocol (SNMP) trap messages, and Traceroute and ping results. SNMP uses UDP to send trap messages based on state variable thresholds. Since UDP is an unreliable protocol, not all trap messages will reach the network management system, resulting in lost observations.

The principle of Occam's Razor is fundamental to network fault localization [23]. Considering the prior probability of any component failing, it is more likely that fewer conditionally independent components can explain observed failures. Considering a typical network component failure probability of 10^{-5} in any given hour [19], the probability that multiple independent components have simultaneously failed decreases significantly with the number of hypothesized simultaneously failed nodes.

Fault localization is further complicated by the existence of errors in the data. These errors include inappropriate prior probabilities, incorrect or inappropriate de-

pendency mappings, and erroneous observations of network state. Prior probabilities used may not be representative of the actual failure rates. Dependency mappings are constructed either through human user input data or by analyzing traffic flow on a network. Observations of network state are determined through client reporting, SNMP traps, or by using probing tools such as Traceroute. Intuitively, any method that relies on human operator input is subject to error. Changes in network configuration are not always updated, resulting in incorrect dependency mappings. Observation nodes are subject to false negative observations, such as lost SNMP packets, and false positive observations, such as spurious symptoms [37] in the network. Errors must be modeled appropriately in any network fault localization approach to identify root causes that best explain observed evidence. Ineffective error modeling can lead to incorrect identification of root causes of network failures, which in turn leads to increased downtime for the network and resources expended to implement failure recovery.

B. FAULT LOCALIZATION ALGORITHMS

Before describing specific fault localization algorithms, this section first summarizes a few core underlying concepts used by recent approaches, including assumptions and heuristics making otherwise intractable algorithms practical for finding identifying faults.

The minimum set-cover problem is known to be NP-Complete [10]. An instance of this problem consists of a finite set X and a family of subsets Y such that each element of X belongs to at least one subset Y. A solution to this problem is the minimum number of subsets Y such that the union of these subsets contain all elements of X. Modeling possible causes of failure as the cover set Y and observations of failure as the set X, as done by one of the studied fault localization approaches [23], identifies the least number of failures to explain the state of observations about failure. Although ideal in its application of Occam's razor, by itself a minimum-set cover

approach lacks a mechanism to leverage prior probabilities and non-binary causal dependencies.

Bayes' rule allows diagnostic inference to reason about causes, given the evidence:

$$Pr(B|A) = \frac{Pr(A|B)Pr(B)}{Pr(A)}.$$

In computer networking, the probability of observing a failure given a component has failed may be estimated or directly measured. That knowledge can be used to leverage the power of Bayes' rule to derive the probability that a component has failed, given the state of an observation.

Many inference methods, such as Bayesian inference, bound the number of simultaneous failures [19]. Reasonable independence assumptions and heuristics can reduce the complexity of this NP-Hard problem [31] with little sacrifice in accuracy. Assumptions about the maximum number of failed components based on the high reliability of networking components also serve to reduce complexity [4,19]. In general, in the networking domain it is reasonable to assume independence between failure causes (e.g. a router fault on one side of a network domain says nothing about whether a cable is cut on the other side) [4, 19]. Additionally, greedy approaches can achieve good results in practice, as does one of the studied fault localization algorithms, SCORE [23].

A survey conducted by Steinder and Sethi, 2004 [37] classifies techniques used in fault localization, dividing them into three broad categories: expert systems, model traversing, and graph theoretic techniques. Expert systems attempt to mimic a human expert to solve problems within a domain. Model traversing techniques represent network entities and their relationships, and then traverse the model graph to correlate alarms and locate faults. Graph theoretic techniques use a fault propagation model to describe entities and conditional dependencies between them in a dependency graph.

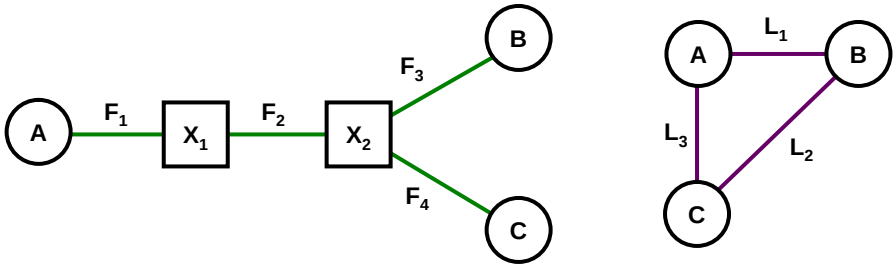
The survey further divides the graph theoretic techniques into five categories. Divide and conquer algorithms cluster dependencies in a dependency graph, then recursively subdivide the clusters containing nodes explaining failure observations. This process continues until a singleton having the highest probability of explaining the failed observations is derived. Context-free grammar approaches represent network components as terminals, and use productions to capture network dependencies. Codebook techniques are represented with a matrix of problem codes that can be used to "look up" the cause given the observed effects. Bayesian network approaches [4, 19] use directed acyclic graphs (DAG) in which nodes represent random variables modeling the state of network elements, and edges representing conditional probabilities [37]. Finally, bipartite causal graph models [23,36,40] use bipartite fault propagation models to represent cause and effect relationships.

A current trend is to model the problem as a DAG having root causes as parentless (root) nodes, observations as childless (leaf) nodes, and dependencies as directed edges in the graph. These edges express conditional probabilities between elements in a network, and allow determining the conditional probability table for each node. This graph structure is also known as a causal graph [37]. The solution approaches using a causal graph typically perform probabilistic inference on the constructed dependency graphs. Most, if not all, network fault propagation models can be transformed and represented with a causal graph.

Root causes to network failure are also known as shared risk groups (SRGs) [25]. Shared risks are typically hardware components that can fail and are represented by the set of nodes that are dependent on the shared risk [23]. In a bipartite causal graph all SRG nodes are root nodes, and members of an SRG set are observation nodes represented graphically by directed dependency edges from the SRG nodes to their member observation nodes. In this dissertation, observation nodes and *observations* are used interchangeably.

Recent approaches exhibiting these techniques include SCORE (non proba-

bilistic), SHRINK, and Sherlock. SHRINK and SCORE use bipartite graph representations of fault propagation models. Sherlock is the first system to expand the approach to use multilevel dependency graphs. In each case the network dependency graph fed to the algorithm is a causal graph. Although SHRINK, SCORE, and Sherlock have many differences, they can all use a bipartite causal graph, and all return a best explanation.



(a) Physical topology (b) IP View

Figure 2.1. Example network.

To illustrate the SCORE, SHRINK, and Sherlock algorithms, consider the simple network depicted in Figure 2.1. Figure 2.1(a) depicts the network physical topology, in which IP routers A, B, and C are connected across fibers F_1 - F_4 and optical cross-connects X_1 and X_2. Each IP router has a an IP link to each other router as shown in Figure 2.1(b). If any of the optical components, fibers, or optical cross-connects fail, the IP routers will detect link failures. The prior SRG failure probabilities are 10^{-4} and 10^{-6} for the fibers and the cross-connect respectively.

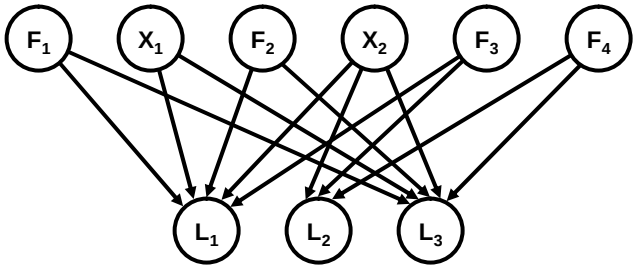


Figure 2.2. Causal Graph for topology in Figure 2.1

The causal graph in Figure 2.2 provides a visual representation of fault propagation for the network in Figure 2.1. Each hardware component that can fail (fibers $F_1 \dots F_4$ and switches X_1 and X_2) are modeled as SRGs, and the IP links ($L_1 \dots L_3$) are modeled as observation nodes. The edge strength of each edge in the graph depicts the probability that an IP link will observe failure given that the SRG has failed. To illustrate, the edge from F_1 to L_1 reflects the probability (unlabeled edges have an edge weight of 1.0) that L_1 will observe failure given F_1 has failed with a probability of 1.0.

1. SCORE

Kompella et al. introduced SCORE [23] in 2005. SCORE applies a greedy minimum set cover technique to perform inference on a bipartite DAG. Two of the strengths of the SCORE algorithm are its inference speed and its adherence to the Occam's razor principle. SCORE does not use probability distributions, however, and therefore may not be the best algorithm to use when probability distribution data is available. SCORE addresses errors through a hit-ratio threshold. This threshold represents the allowable false positive ratio that a hypothesis must not exceed to be a considered as a candidate explanation.

Each potential failure root cause node is represented as a parentless node, and each observation node is represented as a childless node in the graph. The dependencies from parent nodes to child nodes are set to one. Each root cause node has a derived *hit ratio* that reflects the percentage of this node's children that have observed failure, and *coverage ratio* that reflects the percentage of remaining unexplained failures that can be accounted for by the failure of this root cause node. The hit and cover ratios equal $1 - false\ positive\ ratio$ and $1 - false\ negative\ ratio$ respectively. Let O represent the set of observation nodes that have observed failure. Let S_i represents the i^{th} root cause node, and let O_i be the set of observation nodes that will report failure given that S_i has failed. The hit ratio for S_i is equal to $|O_i \cap O|/|O_i|$ and, once computed for each root cause, does not change for the duration

SRG	Links	Hit Ratio	Cover Ratio
F_1	L_1, L_3	0.5	0.5
X_1	L_1, L_3	0.5	0.5
F_2	L_1, L_3	0.5	0.5
X_2	L_1, L_2, L_3	0.67	1.0
F_3	L_1, L_2	1.0	1.0
F_4	L_2, L_3	0.5	0.5

Table 2.1. Hit and Cover ratios calculated for observations $L_1 = down, L_2 = down, L_3 = up$ on the example network in Figure 2.1.

of the algorithm execution. The coverage ratio for S_i is computed using $|O_i \cap O|/|O|$ and is updated with each iteration of the algorithm. With each iteration of the SCORE algorithm the root cause node with the maximum coverage ratio, and having a hit ratio equal to or greater than the input threshold value, is added to a hypothesis vector and the observations associated with the root cause are explained and removed from the set O. The algorithm continues, adding root cause nodes to the hypothesis vector until the observation set O is empty.

Table 2.1 shows an example of using SCORE for the network depicted in Figure 2.1. Consider the scenario where IP links L_1 and L_2 are observed to be down, and L_3 is observed to be up. Intuitively, the cause is most likely the failure of fiber link F_3. On the SCORE algorithm's first pass with a threshold setting of 1.0, only F_3 has a hit ratio of 1.0, so the algorithm adds F_3 to the hypothesis set. The hypothesis completely explains the observations, so the algorithm returns F_3 as the root cause for the failure scenario. In a more complicated scenario, possibly with multiple failures and observation errors, SCORE uses a cost function that considers the number of SRGs in the hypothesis and the threshold setting used for the execution of the algorithm to determine the best explanation.

2. SHRINK

Srikanth Kandula et al. developed SHRINK [19] in 2005 to perform approximate Bayesian inference on a bipartite causal graph. One of the greatest strengths

of the Shrink algorithm is that it can return the probability that a hypothesis has caused the observed failures for each hypothesis considered if the belief values are normalized. Another of SHRINK's main contributions is its robustness. Shrink mitigates potential errors in mapping conditional probabilities and in observation state reporting by reducing each conditional dependency by a noise value, then adding edges with the same noise value to form a complete directed bipartite graph.

The SHRINK model assumes independent failures of root cause nodes and that no more than three SRGs will fail simultaneously in a large network based on the extremely low likelihood of four or more simultaneous failures. Noisy-OR is used to calculate the conditional probability table for a node with multiple parents. The SHRINK algorithm is defined as follows. Let $< S_1, \ldots, S_n >$ denote a hypothesis vector, where $S_i = 1$ if a failure of SRG S_i is assumed, and $S_i = 0$ otherwise. Let $< L_1, \ldots, L_m >$ denote an observation vector, where $L_j = 1$ if a failure of L_j is observed, and $L_j = 0$ otherwise. Given a particular observation vector, the SHRINK algorithm searches through all hypothesis vectors with no more than three assumed failures, and returns those maximizing the posterior probability

$$\underset{<S_1,\ldots,S_n>}{argmax}\, Pr(< S_1, \ldots, S_n > | < L_1, \ldots, L_m >).$$

Consider the example network in Figure 2.1 again. Recall that the causal graph has six optical components mapped to SRGs $F_1 \ldots F_4$, O_1, and O_2. To account for potential database and observation errors a noise value (10^{-4}) is subtracted from the conditional probability of each edge in Figure 2.2, and noisy edges with this same value are added to form a complete bipartite graph. E.g., $Probability(L_1|F_1)$ is 0.9999 while $Probability(L_2|F_1) = 10^{-4}$.

Suppose L_1 and L_2 are down, and L_3 is up. As described above, SHRINK only considers hypothesis vectors with at most three total assumed failures. For this six SRG example SHRINK searches through $\sum_{k=0}^{3} \binom{6}{k} = 42$ hypotheses, with hypothesis vector $< 0, 0, 0, 0, 1, 0 >$ maximizing the posterior probability for the given

observations. SHRINK correctly identifies SRG F_3 (i.e., the failure of fiber link F_3) to be the root cause.

3. Sherlock

Paramvir Bahl et al. introduced Sherlock [4] in 2007. The Sherlock fault localization system uses an inference algorithm called Ferret. One of the strengths of Sherlock's Ferret algorithm is that it can run on a multi-level graph. However, by not considering prior probabilities of SRGs, Sherlock does not adhere to the principle of Occam's razor. One of the main contributions of Sherlock is in directly measuring conditional dependencies in a network to populate its fault propagation model. This data collection mitigates the lack of prior probabilities in its inference model and reduces the risk of human-introduced errors in its SRG databases. Like SHRINK, a noise value is subtracted from all root cause dependencies that affect a network path.

Ferret applies Breadth-First-Search (BFS) to a causal graph, propagating values down to the observation nodes. Ferret can run on a multi-level DAG, and conditional dependencies between root causes are represented by meta-nodes inserted into the graph. In addition to up and down states, Sherlock can compute the belief that a root cause node is in a troubled state: up but experiencing a performance degradation. Like Shrink, up to 3 simultaneous failures are hypothesized using Ferret. Each hypothesis vector is set with a permutation of root causes in either the up or down state. The probabilities at each node are propagated down the graph to the leaf nodes, using noisy-OR computations. The equations used to determine the probabilities that a child node is in different states are

$$P(child\ up) = \prod_j ((1 - d_j) * (p_j^{trouble} + p_j^{down}) + p_j^{up}),$$

$$P(child\ down) = 1 - \prod_j (1 - p_j^{down} + (1 - d_j) * p_j^{down}), \text{and}$$

$$P(child\ troubled) = 1 - (P(child\ up) + P(child\ down)),$$

where d_j represents causal dependency $P(child|parent_j)$, and p_j^{up}, p_j^{down} , and $p_j^{troubled}$ denote the the probability of the j^{th} parent node being up, down, and troubled respectively.

Once all probabilities have propagated to the observation nodes for a hypothesis, the hypothesis score is calculated by taking the product of probabilities of the observation node, where the value used for a node is $P(child\ up)$ if the node reports up, $P(child\ down)$ if the node reports down, and $P(child\ troubled)$ if the node reports troubled. [4] The hypothesis with the highest score is the best explanation of root causes given the observations of state.

Returning to the illustration in Figures 2.1 and 2.2 for the observations L_1 and L_2 down and L_3 up, Sherlock considers $\sum_{k=0}^{3} \binom{6}{k} = 42$ hypotheses. Sherlock assigns edge strengths between a router and path at $1 - 10^{-5}$. Assigning this value to all edges yields a causal graph similar to that used by SHRINK, less prior probabilities and noisy edges. The algorithm returns hypothesis F_3 as the best explanation with a score of 0.9998.

C. CROSS-DOMAIN FAULT LOCALIZATION

Cross-domain fault localization is correlating observations from multiple domains to determine the best explanation for detected faults. Cross-domain, multi-domain, and inter-domain are synonymous terms found in the literature. When data is required from multiple domains to consistently identify the cause of network faults, a cross-domain solution is needed. A study of routing instability found that all parties pointed to another party as the cause in about 10% of the problems [44]. Despite its importance, little work has been done to address fault localization across administrative domains [44].

High-level approaches to model cross-domain fault localization and solutions with limited scope to address the problem have been proposed. Proposed approaches to model collaboration for cross-domain fault isolation are centralized, decentralized,

and distributed strategies [21, 38]. Katzela et al. described three general approaches to cross-domain fault localization collaboration: centralized, decentralized, and distributed [21]. In the centralized approach, a single entity has a global view of the network and performs inference on behalf of all domains involved. The centralized approach introduces a single point of failure for troubleshooting, and is both inefficient and inflexible [35]. A centralized scheme as described by Katzela is akin to the full disclosure approach described in this research, and is therefore not practical for privacy reasons. In the decentralized approach, a central manager oversees all domain managers. When failures affect more than one domain, this central manager coordinates cross-domain collaboration between the domains [21]. In the distributed approach, each network is partitioned into logically autonomous systems. This approach is similar to isolated inference, and includes abstract representations of external root causes that could affect internal observations [21]. A distributed approach is well suited for fault localization when effects from faults do not propagate across network domains [5].

Existing cross-domain approaches tend to either rely on full cooperation from involved domains [18, 35, 38], or on passively monitoring traffic and actively probing to infer network state [2, 33]. In the past, researchers have used routing and update messages, or distributed probing to identify cross-domain failures [48]. Some suggested techniques are based on observing distributed traffic [30, 48]. Distributed fault localization techniques have been identified as an open research problem [37]. A solution to distributed fault localization in hierarchically routed networks has been proposed [38], which is discussed next.

Steinder et al., 2008 presented a cross-domain fault localization approach for hierarchically organized networks that use probabilistic fault propagation models [38]. In this approach, a network manager at the top of a hierarchy oversees and coordinates fault localization for subordinate domains. The approach relies on domains to use probabilistic fault localization algorithms. In the approach, each domain attempts to

localize faults internally first. If the most probable cause of a fault comes from a proxy node representing external domains, the domain manager requests inference from the network manager. The network manager, upon receiving a request for inference from a domain manager, determines the domains transited by the path as reported by the requesting domain. The network manager divides the path into nodes representing the transited domains and the external links between them, and provides each domain the status of the end-to-end path as an external observation. Each domain then correlates this external observation with their internal observations and returns a probability that the root cause resides within the domain. If the most probable explanation is one of the domains, that domain manager is responsible to find the precise root cause.

The cross-domain approach for hierarchical networks does not achieve generality. The approach relies on network domains to fall into a strict hierarchy, such as in either a strictly customer-provider relationship or a hierarchically organized set of domains under the same authority. The approach explicitly looks for errors along a path, meaning that the model must contain all paths through the domains. Finally, each domain must use a probabilistic fault localization algorithm in order to collaborate.

D. NETWORK TOMOGRAPHY

Network tomography, a term first used by Vardi in 1996 [45], uses a limited subset of nodes to monitor a network, and estimate the network status and structure [7, 9, 22, 26]. Network tomography can help to identify routing faults and congestion [9]. However, implementing network tomography on a large scale faces significant computational challenges [7]. Two forms of network tomography in recent literature in include path-level traffic intensity estimation (also known as passive tomography) and link-level parameter estimation (also known as active tomography) [9, 26].

With path-level parameter estimation, nodes inside of a network collect link-level information measurements to estimate path-level parameters [9]. The informa-

tion can then be used to estimate the traffic matrix of a network [26].

In link-level parameter estimation, nodes (typically on the fringe of a network) collect path-level measurements to estimate link parameters [9]. Alternatively, the data may be gathered via probes into the network, hence active tomography [26]. These measurements can be used to characterize a network performance over time [26]. In addition to probing to measure network status, active tomography can be used to reveal a network's hidden structure [9].

With the lack of viable cooperative cross-domain fault localization solutions, active network tomography provides a non-cooperative option for a domain to probe other network domains to gather external evidence. While probing can certainly provide valuable information about the state and structure of another network domain, the information gleaned will not necessarily be of the same quality as information provided by a collaborative approach. Furthermore, there is a growing network security trend to prevent network probing [7, 9], which may reduce the effectiveness of active tomography.

E. PRIVACY CONSIDERATIONS

A passive, or semi-honest, adversary will follow specified protocols and attempt to infer as much information as possible from messages received [47]. In the context of cross-domain fault localization using graph digests, a passive adversary will only attempt to learn sensitive information from a graph digest. Recent work in data-mining uses a semi-honest collaboration model [6, 28]. As in the data-mining work, this research assumes a semi-honest model.

An active, or malicious, adversary will not necessarily follow the protocols, and may take measures to influence the dataset [47]. An active adversary may intentionally induce network faults for the purpose of learning sensitive network properties. This research assumes that collaboration for finding faults is not done with active adversaries.

Graph anonymization is typically done using naive anonymization, in which node labels are simply renamed creating a graph isomorphic to the original, non-anonymized graph [17]. Recent approaches to strengthen anonymization of graphs include random perturbation of edges: performing random edge deletions and insertions [17]. While perturbation helps to enhance privacy of a graph, these perturbations may cause degrade accuracy for the graph [17], and techniques have been proposed to estimate the original data from the perturbed data [20]. Narayanan and Shmatikov successfully identified individual Netflix records, in spite of small data perturbations [29]. Any approach to share inference information for fault localization must take measures beyond simple node anonymization if privacy is a concern.

Recent work to de-anonymize graphs attempt to locate specific nodes in the graph [17]. A common technique to measure privacy for a data set, is to measure k-anonymity as defined by Sweeney [43]. The basic idea of k-anonymity is to create sets of indistinguishable nodes. The cardinality of the smallest of these sets equals the k-anonymization level. Future work to augment the generalized standard deviation metric presented in Chapter III with k-anonymization may strengthen the proposed practical privacy protection approach.

Secure Multiparty Computation (SMC) approaches address performing joint computation in a distributed system where each party reveals no information other than their input and output [27]. Although any polynomial-time multi-party technique can be performed with privacy preservation using SMC, the cost of performing SMC schemes for large-scale models can be too high [6, 27, 28]. As inference models for fault localization are large, a SMC approach to fault localization may not be viable. The framework and approach presented in Chapter III, however, may decrease the size of the models involved sufficiently to enable using SMC.

F. CONCLUSION

As seen in this chapter, there has been recent progress in fault localization. Of particular note, the SCORE (2005) and SHRINK (2005) algorithms both use bipartite causal graphs. SCORE uses an approach that embodies the principle of Occam's Razor, while SHRINK uses Bayesian inference with independence assumptions. The strengths of these two algorithms make them excellent vehicles to test cross-domain fault localization.

Cross-domain fault localization remains an under-researched area, but there have been hints of progress in this research area. The only notable approach found in the literature shows much promise, but is not general in its application.

III. GRAPH DIGEST APPROACH

This chapter presents a framework for cross-domain fault localization that allows any design in the problem space to be evaluated. The chapter provides a set of criteria to explicitly define the two primary requirements of cross-domain fault localization, *realization of inference gain* and *protection of privacy*, and the requirement of *scalability*. The associated metrics for accuracy and scalability are relatively easy to compute and make it possible to experimentally evaluate a design in terms of these criteria. The chapter further provides a specific approach, using graph digests, for use with fault localization models based on causal graphs.

A. GENERAL FRAMEWORK

As discussed in Chapter I, there are three general approaches for diagnosing cross-domain problems. This chapter provides a first formulation of the third approach, whereby domains exchange limited information, e.g., summaries of fault observations, to strike a balance between inference gain and privacy preservation. The crux of the formulation is a set of general metrics to measure the accuracy, privacy protection, and scalability of a given cooperative design.

1. Modeling Inference Gain

A design is useless if the results it produces are not useful for inference. A design cooperative is *inference preserving* if it maintains enough structure to allow successful inference. Ideally, a design achieves the same inference gain as full disclosure.

This research addresses two specific questions regarding the benefits of using a proposed design:

1. What is the change in inference accuracy by using the design for cross-domain scenarios compared to the accuracy achieved when domains perform inference in isolation?

2. What is the decrease in inference accuracy caused by using the design compared to the accuracy achieved when domains collaborate with fully disclosed information?

Question 1 above can be paraphrased as "What is gained by sharing information when troubleshooting a problem?" Question 2 looks at the problem from the other direction: "What is lost by trying to keep some things secret?" If the answer to Question 1 is "a lot" then the design is effective at locating faults. If the answer to Question 2 is "not a lot" then the design is efficient at realizing the potential accuracy gain of cross-domain fault localization.

a. Accuracy Metrics

How is accuracy measured? Consider n domains performing fault localization and let B_T denote the set of actual faults (i.e., the ground truth). Let the best explanation derived by isolated inference be B_i for each domain i. Let the best explanation derived by full disclosure and a proposed design be B_u and B_d respectively. First consider the case of isolated inference. Clearly if $(B_T - (\cup_{i=1}^{n} B_i)) \neq \emptyset$, then the isolated inference results contain false negatives (some faults were not found). The hit ratio [23] is denoted by h_s and measures the percentage of correct results in $\cup_{i=1}^{n} B_i$:

$$h_s = \frac{|(\cup_i B_i) \cap B_T|}{|\cup_i B_i|}. \tag{3.1}$$

Likewise if $((\cup_{i=1}^{n} B_i) - B_T) \neq \emptyset$, then the inference results in isolation have false positives. The coverage ratio [23] (denoted: c_s) measures the percentage of faults in B_T that are correctly identified by $\cup_{i=1}^{n} B_i$:

$$c_s = \frac{|(\cup_i B_i) \cap B_T|}{|B_T|}. \tag{3.2}$$

It is clear that $1 \geq h, c \geq 0$. The ratios of false positives and false negatives are $1 - h$ and $1 - c$ respectively, both relative to B_T. The ratios h and c can each be easily optimized at the expense of the other, which may be overcome by computing the harmonic mean of the two values. The harmonic mean of precision

and recall is also known as F Score [3]. This research proposes to use the harmonic mean α as the criterion to measure how well a digest model preserves inference gain.

The overall accuracy of isolated inference (denoted: α_s) is the harmonic mean of h_s and c_s:

$$\alpha_s = \begin{cases} 0 & \text{if } h_s = c_s = 0 \\ \frac{2 \cdot h_s \cdot c_s}{h_s + c_s} & \text{otherwise.} \end{cases} \tag{3.3}$$

The value of α_s ranges from 0 (zero accuracy) to 1 (perfect inference). Intuitively, a small α_s value indicates a need for cross-domain coordination.

The accuracy using full disclosure (*undigested* graphs) is calculated by

$$\alpha_u = \begin{cases} 0 & \text{if } h_u = c_u = 0 \\ \frac{2 \cdot h_u \cdot c_u}{h_u + c_u} & \text{otherwise,} \end{cases} \tag{3.4}$$

where

$$h_u = \frac{|B_u \cap B_T|}{|B_u|} \text{and } c_u = \frac{|B_u \cap B_T|}{|B_T|}. \tag{3.5}$$

The accuracy of a proposed design is calculated by

$$\alpha_d = \begin{cases} 0 & \text{if } h_d = c_d = 0 \\ \frac{2 \cdot h_d \cdot c_d}{h_d + c_d} & \text{otherwise,} \end{cases} \tag{3.6}$$

where

$$h_d = \frac{|B_d \cap B_T|}{|B_d|} \text{and } c_d = \frac{|B_d \cap B_T|}{|B_T|}. \tag{3.7}$$

Without special consideration, a failure hypothesis involving $x > 1$ indistinguishable faults will result in adding x faults to the best explanation every time, adversely impacting the hit ratio of the hypothesis. These faults are combined into a single fault to calculate the scores α_u, α_d, and α_s. Consolidating indistinguishable faults is consistent with the SCORE fault localization algorithm [23].

To quantify the inference gain A from using a design (i.e., to answer question 1 above), this research proposes to compute the difference between its inference accuracy and the accuracy achieved by domains in isolation:

$$A = \alpha_d - \alpha_s. \tag{3.8}$$

The value of A ranges from -1.0 to 1.0. A positive score means that the design improved fault localization, a score of 0.0 means there was no improvement, and a negative value means that using the design was worse than isolated inference. For example, suppose $B_T = \{S1, S4\}$, $\cup_i B_i = \{S2, S5\}$, and $B_d = \{S1, S5\}$. Then $h_s = c_s = 0$ and $h_d = c_d = 0.5$. Thus, the inference gain A equals 0.5 for this case.

Similarly, this research proposes to measure the cost to privacy protection C (i.e., to answer question 2 above) with the metric

$$C = \alpha_u - \alpha_d, \tag{3.9}$$

where C ranges from -1.0 to 1.0, with a larger value indicating a higher cost [1]. Continuing with the example above, C would be 0.5 if the full disclosure approach achieves perfect accuracy, i.e., $B_u = B_T$, which implies $\alpha_u = 1.0$.

Note that a design that only shares limited information may require dramatically less computation as compared to the full disclosure approach. In other words, the design is much more scalable. Section 3 discusses how to quantify this benefit.

In support of the hypothesis of this dissertation, experiments showed that a prototype design generally achieves better accuracy than isolated inference. To test this hypothesis, the null hypothesis H_0 used was: a prototype design is no better than isolated inference. H_0 is equivalent to $\alpha_d = \alpha_s$, which is just $A = 0$. The alternative hypothesis H_1 was that the design is more accurate than isolated inference, or $A > 0$.

2. Modeling Privacy Preservation

In developing criteria for privacy preservation this section first presents a metric to measure how much information a design discloses that would otherwise remain undisclosed. Recognizing that this ideal metric may not be practical, Section B

[1] Intuitively, C should range from 0.0 to 1.0

presents practical metrics to measure information disclosure from using a specific design (using a graph digest approach).

Before measuring privacy preservation, first the information that needs to be protected must be established. This research defines a *sensitive property* as a piece of information a domain manager considers private. Ideally, shared information should not help to reveal any sensitive properties. Information about sensitive properties should never be distributed unless permitted by a domain's local security policy. Specific sensitive properties will vary between domains and may include bottlenecks, customer information, peering agreements, and many other characteristics. Furthermore, a collection of exchanged information from a domain over time should not aid in deriving the sensitive properties.

Shannon said that "perfect secrecy" is achieved when the *a priori* probability is equal to the *a posteriori* probability for message traffic deciphering by an adversary [32]. The same concept applies sharing inference information. One has to assume that an adversary has some domain knowledge, has passive access to externally observable information, and can infer some level of knowledge about a distribution over time. As discussed in Chapter II, this research assumes a semi-honest model.

This research explores information theory, which overlaps several technical fields, to address the privacy preservation issue. Entropy, typically measured in bits, is foundational to information theory. Entropy measures uncertainty about a probability distribution. The entropy $H(X)$ of the random variable X with a probability mass function $p(x)$ is defined by

$$H(X) = -\sum p(x)\ log_2\ p(x).$$

Information theory provides a means to reason about entropy between two distributions. [11]

Using an information theoretic approach, the relative entropy, or Kullback Leibler (KL) distance [11], between a probability mass function of the random variable representing an adversary's belief about a sensitive property's true value without

shared information and a probability mass function after receiving shared information measures the privacy loss due to implementing a cooperative design. Consider a sensitive property that can be modeled by a discrete random variable X. Let $p(\ )$ represent the probability mass function representing an adversary's belief about this sensitive property conditioned by externally available information. Let $d(\ )$ represent the probability mass function representing the adversary's belief further conditioned by shared information. The KL relative entropy equation is

$$KL(p(\ )||d(\ )) = \sum_x d(x)\log_2\frac{d(x)}{p(x)}. \tag{3.10}$$

In the best case this distance will equal zero for each sensitive property in a domain, meaning that the information about a sensitive property is unchanged after sharing information. Even if the entropy is reduced for a sensitive property, the entropy of $d(x)$ may remain sufficiently high to protect the privacy of the property. Ultimately, the resultant entropy of $d(x)$ and not the amount of entropy lost as given by Equation 3.10, indicates the level of privacy protection for a sensitive property.

If prior and posterior probability distributions modeling an adversary's belief about a sensitive property can be derived, the relative entropy Eq. (3.10) can be used to evaluate the privacy protection for a property. Although the KL distance appears to be a perfect measure of privacy preservation, it is extremely difficult to apply in practice as computing the KL distance requires knowledge of the prior probability distribution the adversary uses (explicitly or implicitly) to guess a secret.

3. Modeling Scalability

Consider the SHRINK algorithm, which achieves polynomial time inference by assuming no more than 3 concurrent SRG failures [19]. The algorithm still must consider $\binom{n}{1} + \binom{n}{2} + \binom{n}{3}$ hypotheses [2] with n here denoting the total number of SRGs. The computational complexity for SHRINK is $O(n^4)$. Clearly, by compressing

[2] After abstracting away the null hypothesis and the "not in the model" hypothesis

information a cooperative design will reduce the magnitude of elements in a model (e.g. the number of SRGs n in the SHRINK model), resulting in far fewer hypotheses to consider vs. full disclosure. As a result, such a design is intuitively more scalable in terms of inference running time.

This research proposes a direct measurement of inference running times to evaluate scalability. Let t_u and t_d represent the recorded average running times for the full-disclosure and a proposed design respectively. The metric E to quantify the scalability improvement is defined by

$$E = log_{10}\left(\frac{t_u}{t_d}\right). \tag{3.11}$$

Thus, E measures the order of magnitude of reduction in inference time gained by using a cooperative design as compared to full disclosure. A logarithmic measurement is used for E to clearly present the order of magnitude difference in the running time. A value for E much greater than 0 reflects significant savings in inference time by using the proposed design, a value close to 0 reflects little or no savings, and a value less than 0 means that the design performed slower than full disclosure inference.

B. GRAPH DIGEST APPROACH

As discussed in Chapter II, recent intra-domain approaches use graphical models to represent dependencies in a network, particularly the causal relationships between hardware failures and observed anomalies. These models (also called inference graphs), enable inference algorithms to determine those failure scenarios best explaining observed anomalies. In practice, faults often propagate across network domain boundaries, depriving intra-domain algorithms of critical information required for accurate inference. This research addresses the problem by sharing summarized intra-domain models, called *graph digests* or simply *digests* in this research, between domains. A graph digest is created to reflect a failure scenario and captures cross-domain dependencies while hiding internal details.

A cross-domain inference model based on graph digests can be formally defined as follows. Consider n network domains:

- G_i is the inference graph for the i_{th} domain.
- f is (ideally) a one-way transformation on G_i implementing a privacy policy. $f(G_i)$ is called the *inference graph digest*, or simply *digest*, for G_i.
- $\mathcal{G}^j = \left(\biguplus_{i \neq j}^{n} f(G_i) \right) \uplus G_j$, where j is a domain performing cross-domain inference and $\uplus$ is a model-specific union. $\mathcal{G}^j$ is the cross-domain model integrating the digests from all the other domains with domain j's undigested graph. Now, domain j may use an existing algorithm such as SHRINK to perform inference over $\mathcal{G}^j$.

Before a practical graph digest design can be implemented, interoperability standards must be developed. Domains using different inference methods can potentially use a digest approach if standards are implemented and adhered to. Items to be standardized include data types and attributes as well as cross-domain management structures such as centralized, distributed, iterative, etc. Translation procedures are needed in order to convert between models. This research defines a *shared attribute* as a physical entity or logical concept modeled in two or more fault propagation inference graphs, and that has the same semantics in each graph. Shared attributes serve as the glue that allows different models to be joined. For example, a shared attribute that models the event that packets flow across a peering link between two domains may be modeled as a root cause in one domain's model and a dependent observation in the other. In order to create a domain digest to connect to another domain's fault propagation inference graph, shared attributes must be identified and agreed upon.

The process for creating a graph digest is outlined in Figure 3.1. When a fault is detected, if isolated inference does not find the root cause participating domains agree on which domain will perform inference using their undigested inference graph (graph G_j above). Each other domain creates a digest, if required, using the process in Figure 3.1. The final decision in the graph digest creation process places responsibility

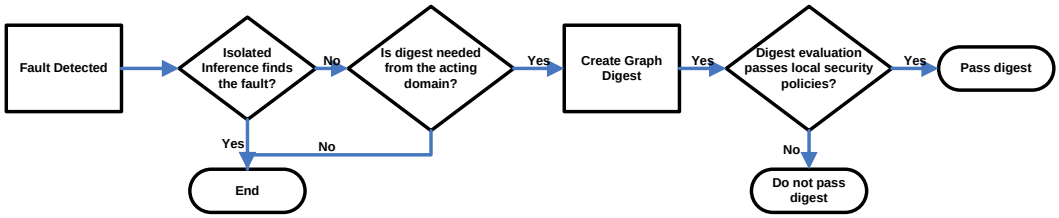


Figure 3.1. Process to create a graph digest

on the domain creating the digest to ensure compliance with local security policy. This verification step places a human in the loop to enforce privacy of properties deemed sensitive by the network administrator.

1. Practical Privacy Protection Metrics

The KL Distance metric for privacy is applicable when realistic distributions representing an adversary's belief about a sensitive property can be constructed. Unfortunately, deriving accurate probability mass functions about a sensitive property in a domain, particularly from an adversary's perspective, may not be possible. This research explores a more pragmatic approach: characterize the effectiveness of various attacks against a digest to learn specific sensitive properties about the digest's source domain. Specifically, the research provides a systematic method for experimentally evaluating attacks against a causal graph.

a. Modeling a Causal Graph Attack

The focus of this research is on developing a general evaluation methodology, not on developing the most effective attacks on causal graphs. An exploration of different sensitive properties to demonstrate privacy using the KL Distance did not find a sensitive property with a meaningful probability distribution function.

As a practical approach, this research explored learning a domain's topology (routers, switches, physical and VPN links, etc.) from the network's inference graph. Not surprisingly, the literature search did not uncover previous work addressing attacks on causal graphs. Once portions of a network topology have been learned, sensitive property measurements can be taken on the constructed topology.

b. Modeling Attack Effectiveness

There are many properties that a network domain administrator may consider sensitive. However, relatively few of these can be inferred from an inference graph for the domain. For example, properties such as detailed customer information or operating system details, are most likely abstracted away from the graph. Four example sensitive properties that could be inferred from an inference graph and used to evaluate privacy protection are:

- Domain network diameter
- Number of routers in a domain
- Degree of the node with the highest degree in a domain
- Internal reachability between a pair of visible gateways

This research proposes to use the following statistical metrics to model the effectiveness of an attack against a sensitive property.

- Root mean square error ($rMSE$).

  Let $X = \{x_1, x_2, ..., x_m\}$ represent the collection of samples for a set of m scenarios where the property has a fixed true value of P. The $rMSE$ for that scenario set is defined by

  $$rMSE = \sqrt{E((X-P)^2)} = \sqrt{\sum_{i=1}^{m}(x_i - P)^2/m}. \tag{3.12}$$

  The interpretation of $rMSE$ is straightforward: if the $rMSE$ value is large relative to the true value P, the attack is considered unsuccessful.

- Generalized standard deviation ($gSTD$).

  Usually the standard deviation, like $rMSE$, should be defined with respect to a set of scenarios where the property's true value is fixed. The definition is generalized to consider samples from all scenarios used in an evaluation. Let $\{x_1, x_2, ..., x_M\}$ represent the collection of samples for all M scenarios. The $gSTD$ is computed like a usual standard deviation by

  $$gSTD = \sqrt{E((X - E(X))^2)}. \tag{3.13}$$

The $gSTD$ has a desirable feature: it captures how well the attack algorithm tracks the fluctuation in the true value of the property. This point will be further articulated in Chapter VI. The attack is considered not effective if $gSTD$ is small relative to the sample mean $E(X)$. For this reason, $gSTD$ can be viewed as a good indicator of the KL distance.

C. CONCLUSION

This chapter covers the two most significant contributions of the research: first, a general framework, and second, a graph digest approach.

The general framework provides solid metrics to evaluate any cooperative design in the design spectrum defined by the competing requirements of accuracy and privacy, and scalability. These metrics provide a transparent and rigorous way to address the core issues for cross-domain fault localization and evaluate any such design.

In the graph digest approach, domains provide abstract representations of their fault propagation models, coupled with evidence, to strike a balance between accuracy and privacy. By distilling the shared information down to a collection of nodes and edges, the approach intuitively reduces the size of a model as compared to the full disclosure approach.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. EVALUATION METHODOLOGY

This chapter presents the methodology used to evaluate the framework presented in Chapter III. A prototype algorithm for creating graph digests from bipartite causal graphs was developed for evaluation. This algorithm is used to construct digests for numerous failure scenarios across a range of realistic network topologies, and then evaluated in terms of inference accuracy, privacy preservation, and scalability as defined in Chapter III. The evaluation was shaped to consider scenarios with inherent cross-domain characteristics.

The test topology selection process was motivated by several factors. Most importantly, to satisfy the generality of the approach, the test topologies represent a wide range of realistic networks. It is well-known that there is a lack of topologies available for research efforts [34]. Instrumented networks may not contain enough variety in topology to represent networks in the general case, and almost certainly would not yield reproducible results. Furthermore, as this research is about fault localization across network domains, collected data from instrumented networks would not necessarily contain sufficient numbers of samples for meaningful results. The Naval Postgraduate School is not a member of either the Abilene or GEANT projects, and any collected cross-domain data from this project, if it exists, is not publicly available. Given the need for generality and lack of available suitable test topologies, topologies containing realistic topology components (atoms) were constructed. To provide empirical evidence of the applicability to real network topologies, an additional topology based on the Abilene [1] network was used.

To further evaluate the generality of the approach, experiments were conducted by performing inference with both the SHRINK and SCORE algorithms using six different synthetic topologies: two types of relationship between domains, and each relationship has three topologies (small, medium, and large). Both provider-customer and peer-peer relationships were modeled since autonomous systems typically peer

using one of these two relationships [42], and so the generality of shared attributes could be demonstrated. Additional experiments were performed using a topology based on the Abilene network in a provider-customer setting. For each topology, data was collected for *single* and *double* failure scenarios. If an observation node could exist in another domain that provides evidence about an SRG, this research defines that SRG as a *cross-domain SRG*. Failure scenarios were generated randomly, but in order to favor scenarios requiring cross-domain fault localization, failure selection was constrained such that at least one failure in a scenario must be a cross-domain SRG. All single failure scenarios that satisfy this constraint were evaluated. There are a total of 9 such scenarios in each provider-customer setting. In the peer-peer setting there are 24, 42, and 68 such failures in the small, medium, and large topology respectively. Three data collection cycles of fifty failure scenarios each for the double failure scenarios were executed, yielding 150 distinct double failure scenarios for each of the small, medium, and Abilene-based topologies. For both of the large topologies, two collection cycles of twenty-five failure scenarios were executed, resulting in fifty distinct double failure scenarios.

A variation of the decentralized collaboration model (Chapter II) was selected for digest exchange. In the implemented model, each participating domain passed a digest to a single domain, which then performed inference on behalf of all of the participating domains. The model intuitively provides the best choice for the provider-customer domain relationship. With many customers, the provider network domain has shared attributes with each customer, while customers may not have shared attributes with each other. In each failure scenario the domain identified as Domain 1 performed inference, adding a digest from Domain 2 to the Domain 1 causal graph.

Detailed descriptions of the models used with early versions of the prototype algorithm and attack heuristic are documented in previously published incremental steps of this research [15, 16].

```
createBipartiteDigest(G)
 1: Add node L_new to G
 2: for all SRG S_i ∈ G
 3:    if (for all edges (S_i, L_j) ∈ G, L_j is up)
 4:       then Prune S_i and its edges (S_i, L_j)
 5:    else
 6:       Collect edges (S_i, L_j) ∈ G such that L_j is up
 7:       if At least one such edge exists
 8:          Add edge (S_i, L_new)
 9:       Prune collected edges (S_i, L_j)
10: Remove all isolated observation nodes L_i
11: for all SRG S_x, S_y ∈ G
12:    if S_x and S_y are indistinguishable
13:       Aggregate S_x and S_y into S'_x such that S'_x = S_x ∪ S_y
14: Rename all SRGs that are not shared attributes
15: Rename all Observation nodes other than L_new
```

Figure 4.1. Algorithm for computing a digest from a bipartite causal graph G.

A. DIGEST ALGORITHM

As the target of evaluation, a prototype digest creation algorithm was created. The algorithm uses simple techniques, such as node and edge pruning, partial evaluation, aggregation, and node renaming. Information such as prior probabilities and conditional probabilities have been anonymized by setting all respective values to the same strength. The algorithm originally used Noisy-OR to combine edges in the digest causal graph directed to "up" observation nodes. Using Noisy-OR helped to preserve inference information about these observation nodes that are conditionally dependent on the SRG being evaluated. However, this practice was found to be very revealing about a domain network topology since the edge strength indicated the number of neighbors a device has. Therefore, Noisy-OR was replaced with logical OR in the implementation of the algorithm. See Figure 4.1 for detailed pseudo code of the digest creation algorithm.

The focus of this work is on the evaluation methodology and hence the development of potentially more effective digest creation algorithms is left to future work.

If a rather simplistic digest creation algorithm performed promisingly, however, it would be reasonable to conclude positively about the feasibility of cross-domain fault localization when more polished techniques are used.

B. PROVIDER-CUSTOMER TEST TOPOLOGIES

This section introduces the provider-customer test topologies. In a provider-customer relationship, one domain (the provider) provisions network backbone connectivity to a second domain (the customer). In many cases, the provider's physical topology (e.g., SONET connections multiplexed on fiber) is not observable by the customer. The customer only sees IP connections entering the edge device on one side of the provider's cloud and exiting on the other. Sometimes only a core router is visible. In any case, many sources of faults are not visible to the customer. Furthermore, configuration problems on either the customer's or the provider's side may result in faults that are not readily observable by both parties.

For the provider-customer topologies, with the assumption that failures are not total in the provider network and individual IP flows are not instrumented for fault detection by the provider, observations were denied about customer flows to the provider.

1. Physical Topologies

Each topology simulates a provider-customer network setting in which a customer transits the provider domain using three leased circuits. The small topology depicted in Figure 4.2, is loosely based on the topology used by the authors of SHRINK [19]. The provider network (Domain 1 in Figure 4.2) consists of Optical Digital Cross Connect switches and fiber links to transit customer traffic. The customer in the evaluation leases three optical circuits that transit the fiber mesh as depicted in Figure 4.3. Additionally, several VPN tunnels are modeled in the customer topology, shown in Figure 4.4, to explore the effects this realistic networking practice has on

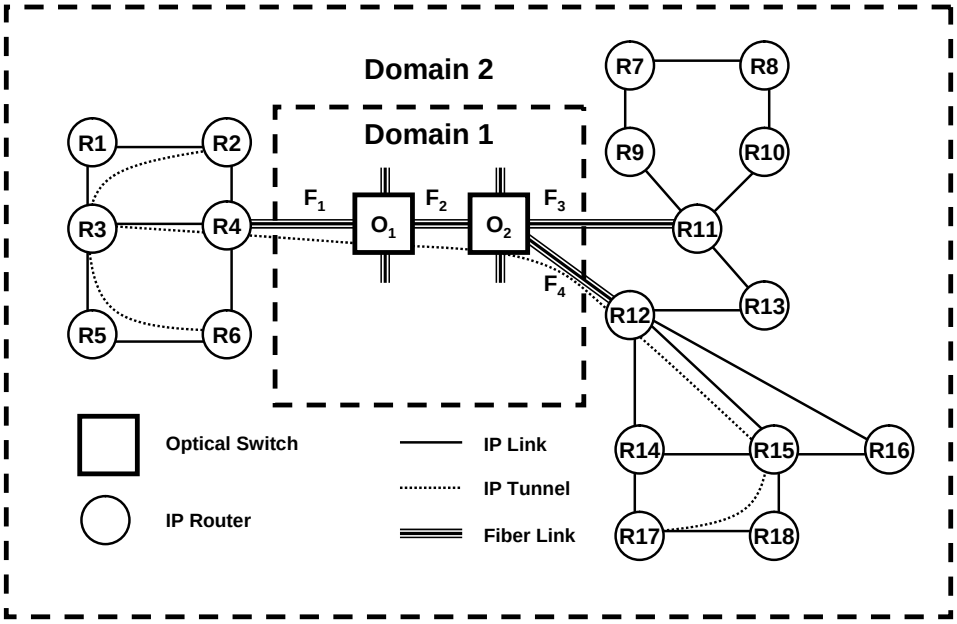


Figure 4.2. Provider-Customer Small Physical Topology.

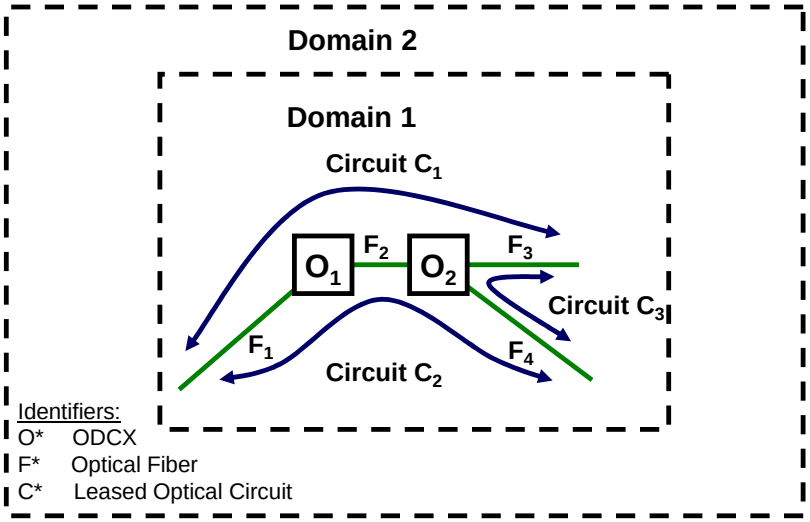


Figure 4.3. Provider-Customer View of Service Provisioning.

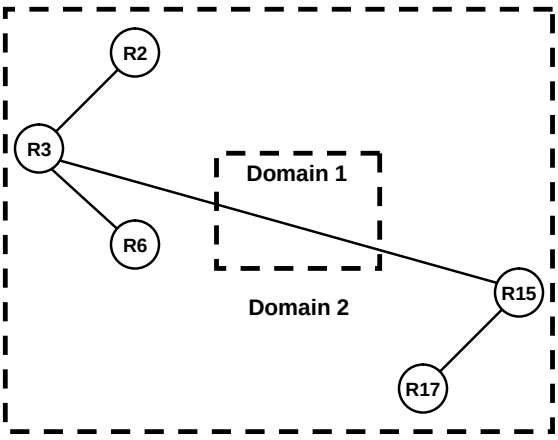


Figure 4.4. Provider-Customer VPN Overlay.

the framework. The study focuses on finding cross-domain faults that occur between the provider domain and one of its customers (Domain 2 in Figure 4.2).

The customer topology grew on each side, adding sub-components to reflect realistic network topology elements to create the medium (Figure 4.5) and large (Figure 4.6) network topologies. The small, medium, and large customer network domains

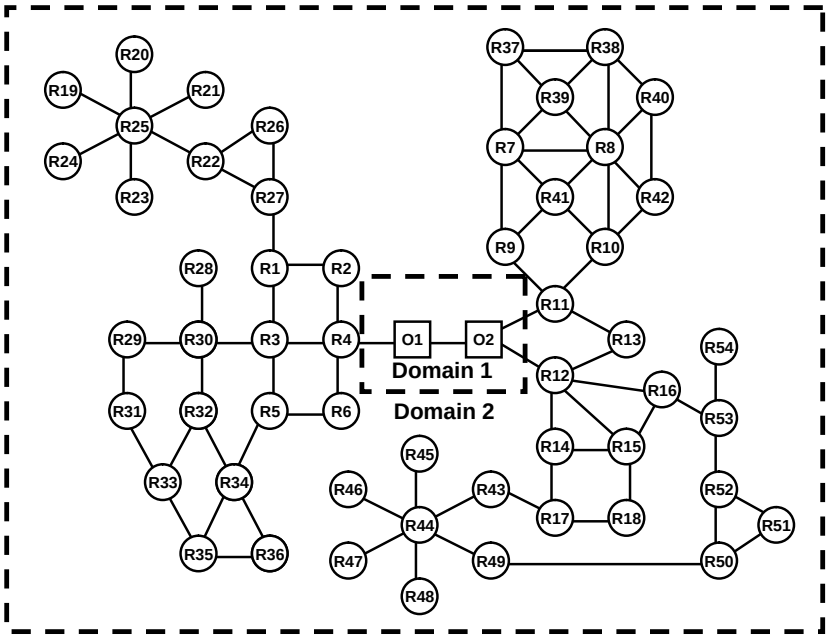


Figure 4.5. Provider-Customer Medium Physical Topology.

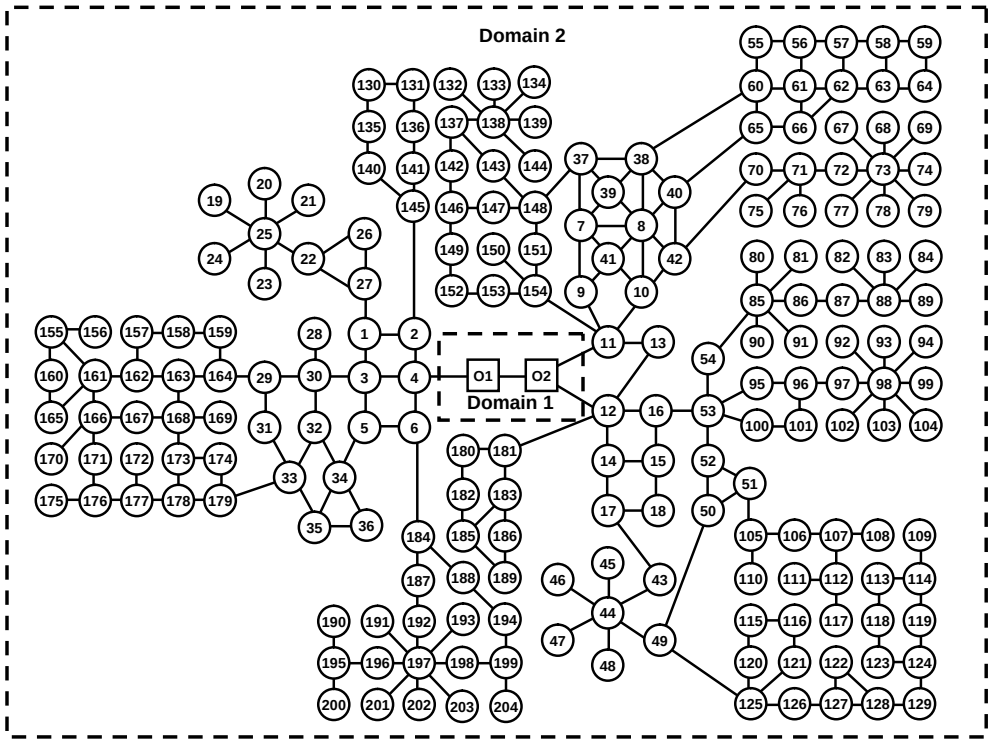


Figure 4.6. Provider-Customer Large Physical Topology.

have 18, 54, and 204 routers respectively. The sub-components in the expanded networks have varying properties, such as node degree and distance between elements in the domain, and are connected in mesh, star, ring, and *ad hoc* topologies. The medium and large network topologies use the same provisioning (Figure 4.3) and customer VPN tunnels (Figure 4.4) as in the small topology.

2. Modeling

To illustrate how a causal graph models fault propagation and how graph digests are used, a detailed description of inference using the graph digest approach is provided next using the small provider-customer physical topology (Figure 4.2).

As illustrated in Figure 4.2, Domain 1 has two optical cross connect switches (O_1 and O_2) and four fiber links ($F_1 \ldots F_4$) as SRGs. In Domain 2 (the customer domain) each router ($R_1 \ldots R_{18}$) and point-to-point link between adjacent routers (e.g., $R_1 - R_3$) are modeled as an SRG. Every SRG failure in the customer domain

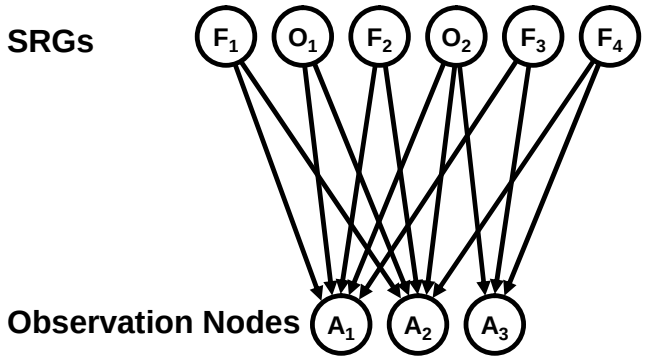


Figure 4.7. Domain 1 causal graph with respect to Domain 2.

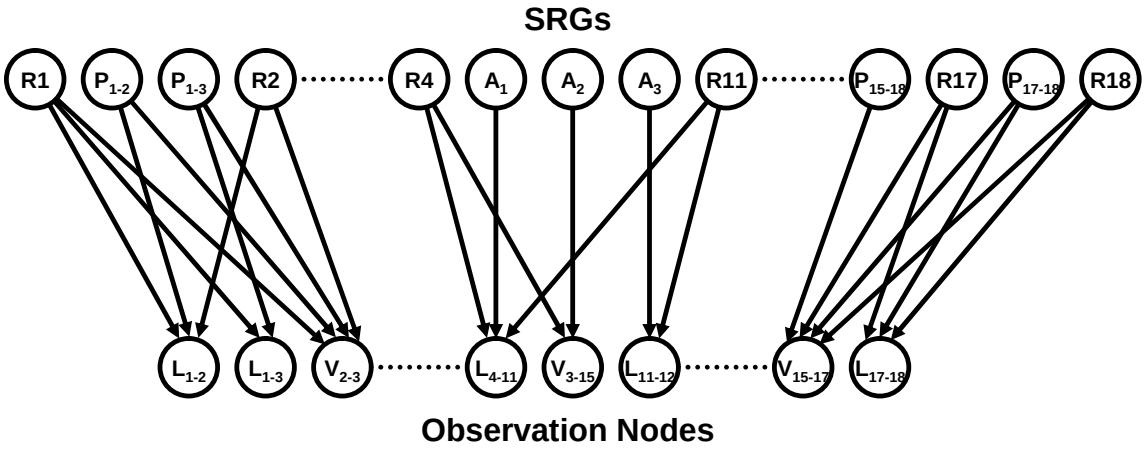


Figure 4.8. Domain 2 causal graph for small topology.

generates observations about the failure. The following observation nodes are modeled in Domain 2: the IP connections between each pair of adjacent routers; the 3 internal VPN tunnels ($R2-R3$), ($R3-R6$), and ($R15-R17$); the cross-domain IP connections ($R4-R11$), ($R4-R12$), and ($R11-R12$); and the cross-domain VPN tunnel ($R3-R15$). The three leased circuits underlying the cross-domain IP links serve as the shared attributes for this setting, with Domain 1 modeling the shared attributes as observation nodes ($A_1 \ldots A_3$ in Figure 4.7), and Domain 2 modeling them as SRG nodes. There are nine cross-domain SRGs from both domains (O_1, O_2, $F_1 \ldots F_4$, R_4, R_{11}, and R_{12}) in the customer-provider setting.

As the provider domain (Domain 1) would have many cross-domain SRGs

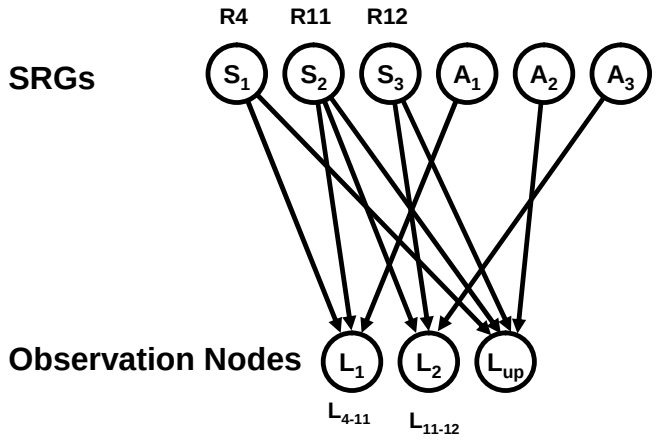


Figure 4.9. Domain 2 digest for small topology.

for different customers and a contractual obligation to provide transit service, the provider domain was selected to perform inference on behalf of its customers for the graph digest approach. For each of the failure scenarios, the customer domain generated a digest for inference by the provider domain. The Domain 2 small topology causal graph is presented in Figure 4.8. The routers $R1 \ldots R18$, the point-to-point links P_{x-y} where x and y are the pair of adjacent routers Rx and Ry, and the shared attributes $A_1 \ldots A_3$ are identified as the SRGs. The observation nodes are the IP links between the routers $L_{x,y}$ and VPN tunnels $V_{x,y}$, where x and y designate the routers on either end of the links or tunnels.

The Domain 2 digest created after observing connection failures L_{4-11} and L_{11-12} is depicted in Figure 4.9. The SRGs $R4$, $R11$, and $R12$ have been anonymized as $S_{1\ldots3}$, and IP links L_{4-11} and L_{11-12} as L_1 and L_2. Only the special observation node L_{up} observes an "up" state and all other observation nodes report a "down" state. All SRG prior probabilities are set to a uniform value; likewise all conditional dependencies (the edges) have a uniform value.

An ad hoc node collapsing methodology is used to form a union between the causal graphs, which starts by merging the shared attributes from each causal graph. Next, each observation node inherits all conditional dependencies from all shared

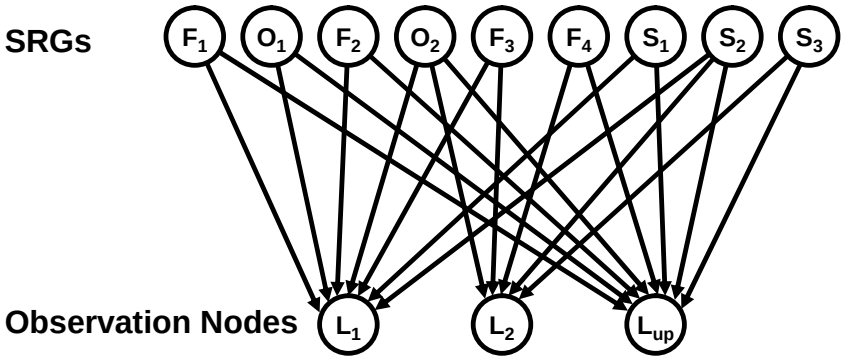


Figure 4.10. Union of causal graphs.

attributes on which the observation node is dependent (e.g., if an edge exists from a shared attribute to an observation node, then all edges into the shared attribute from an SRG are copied to that observation node). Finally, the shared attribute nodes are removed. As an example F_1 has an edge to A_1 in the Domain 1 causal graph (Figure 4.7) and A_1 has an edge to L_1 in the Domain 2 digest (Figure 4.8), thus F_1 gains an edge to L_1 in the causal graph union. The model-specific union of the Domain 1 causal graph with Domain 2's digest is shown in Figure 4.10. In this sample scenario SHRINK and SCORE each return F_3 as the best explanation for both the full disclosure and graph digest approaches.

There may be information loss with the transitive method of inheriting conditional dependence described above. Consider the observation node L_{4-11} in Figure 4.8. This node is conditionally dependent on three SRGs in the Domain 2 graph. The same node, represented by L_1 in Figure 4.10, is conditionally dependent on seven SRGs after creating the union of the Domain 2 digest and the undigested Digest 1 causal graph. Depending on the failure scenario, a different number of SRGs may populate the conditional probability table (CPT) for observation node L_{4-11}. Since the prior probabilities of the SRGs and the conditional dependencies have been set uniformly in this example scenario, the values in the table are distorted and information is lost. If the correct distributions were learned over time, or intentionally disclosed, the CPTs for the shared attributes could be populated with the correct

distributions.

In the provider-customer topologies (Figures 4.2, 4.5, 4.6) F_1, F_2, and O_1 are indistinguishable to SHRINK and SCORE. Statistically, $\frac{1}{3}$ of the randomly generated failure scenarios will contain the failure of one of these three components. As discussed in Chapter III, these nodes are combined into a single SRG to calculate the α_u, α_d, and α_s scores.

C. PEER-PEER TEST TOPOLOGIES

The second class of topologies considers two network domains with a peer-peer relationship. In this relationship, each domain provides connectivity to its customers, and neither domain provides Internet connectivity to the other [42]. The two domains share multiple peering points and web service connections. These web service connections represent monitored IP connections of interest between pairs of servers hosted in different domains. Ownership of the shared links and hosting of the services may be equally distributed between the two domains. IP link and web service failures are fully visible, and device failures are considered total - an SRG failure causes an observable failure event.

1. Physical Topologies

A similar process to create the provider-customer topologies was used to create the peer-peer topologies. incorporating realistic network domain subcomponents. The small physical topology is presented in Figure 4.11. The peer-peer domains in the small topology have two peering points ($R4 - R11$) and ($R6 - R17$). The shared attributes $A1$ and $A2$ model the event that the cross-domain connection is live.

There are five web services, $W1 \ldots W5$, with cross-domain dependencies as shown in Figure 4.12. For each web service with a cross-domain dependency, one domain models ownership of the web service and the other domain models a dependency on that service. The shared attributes $A3 \ldots A6$ model the event that the servers can

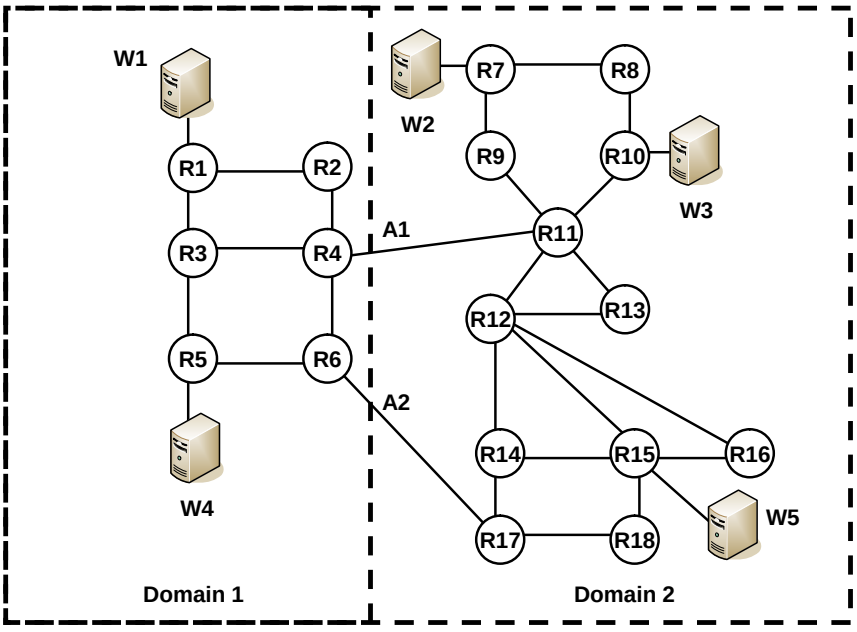


Figure 4.11. Peer-Peer Domains Small Physical Topology.

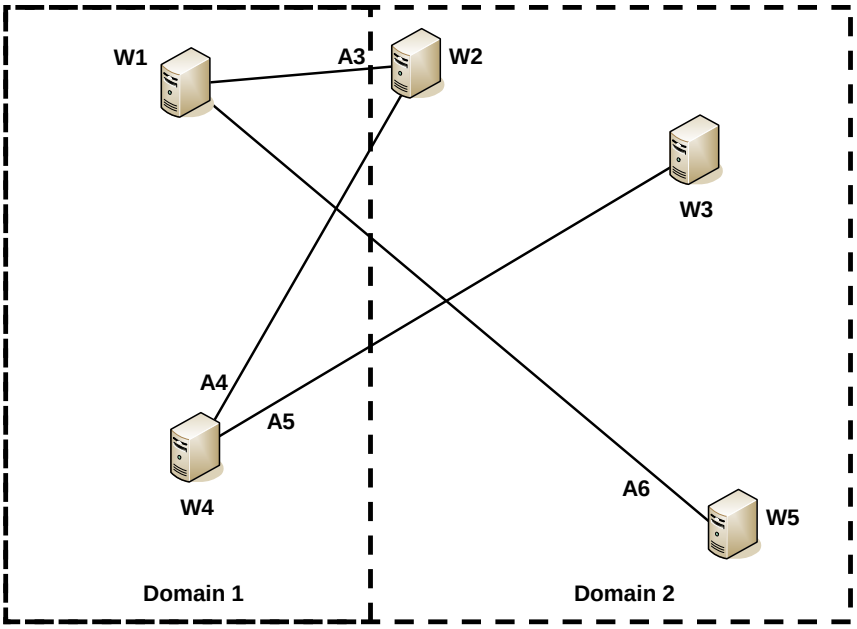


Figure 4.12. Peer-Peer Domains Small Services View.

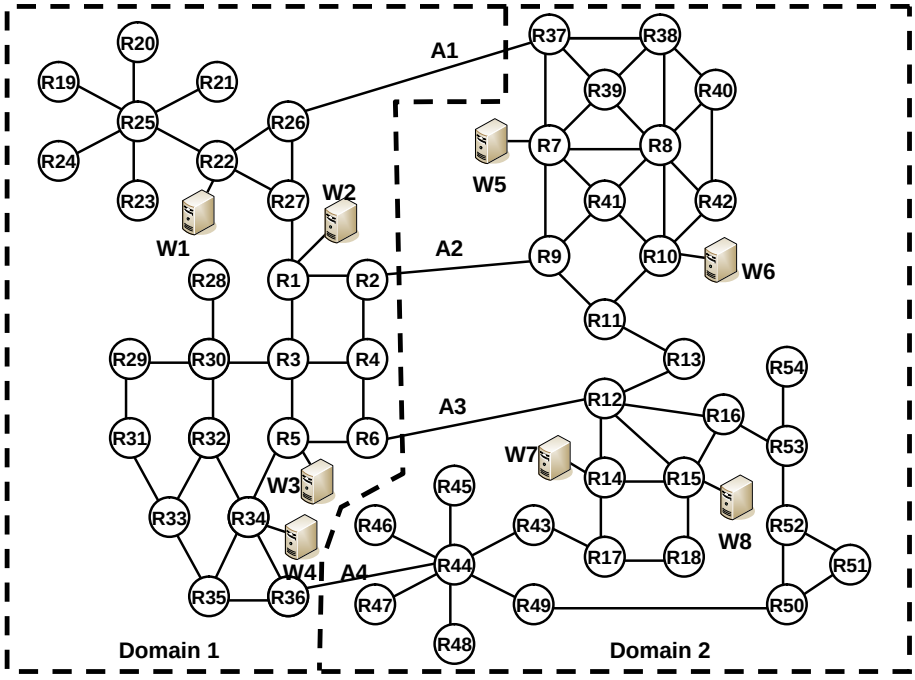


Figure 4.13. Peer-Peer Domains Medium Physical Topology.

reach each other along the shortest path for each pair of dependent services.

The medium topology (Figure 4.13) is modeled with four peering points and eight web service connections (Figure 4.14), and the large topology (Figure 4.15) with eight peering points and sixteen web service connections (Figure 4.16). The treatment of the peering points, cross-domain web services, and shared attributes follows the same reasoning and implementation as discussed for the small domain topology.

2. Modeling

The SRG and observation nodes are modeled as in the customer-provider setting, and use the same notation. The set of cross-domain SRGs, from which each failure scenario must have a failed component, contains every peering point router and link, and every router and link on the shortest path between the servers for each web service dependency. A total of 24, 42, and 68 cross-domain SRGs were identified in the small, medium, and large topologies, respectively.

There are two types of shared attributes for this scenario. The first type

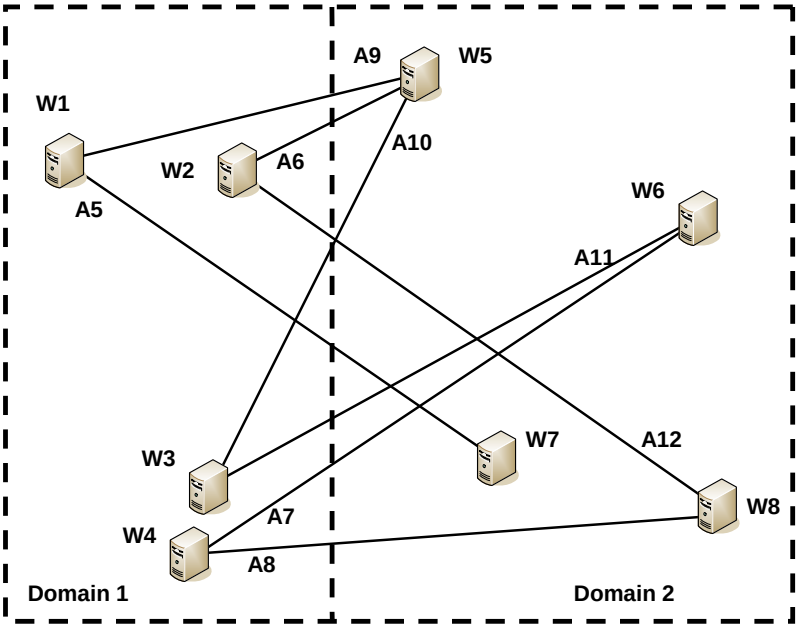


Figure 4.14. Peer-Peer Domains Medium Services View.

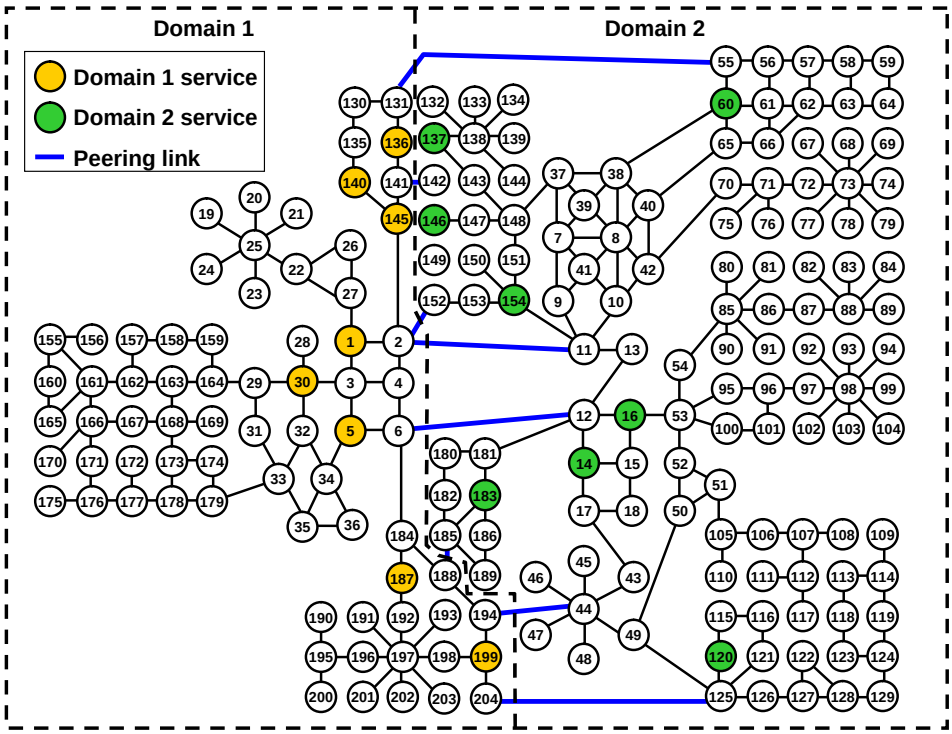


Figure 4.15. Peer-Peer Domains Large Physical Topology.

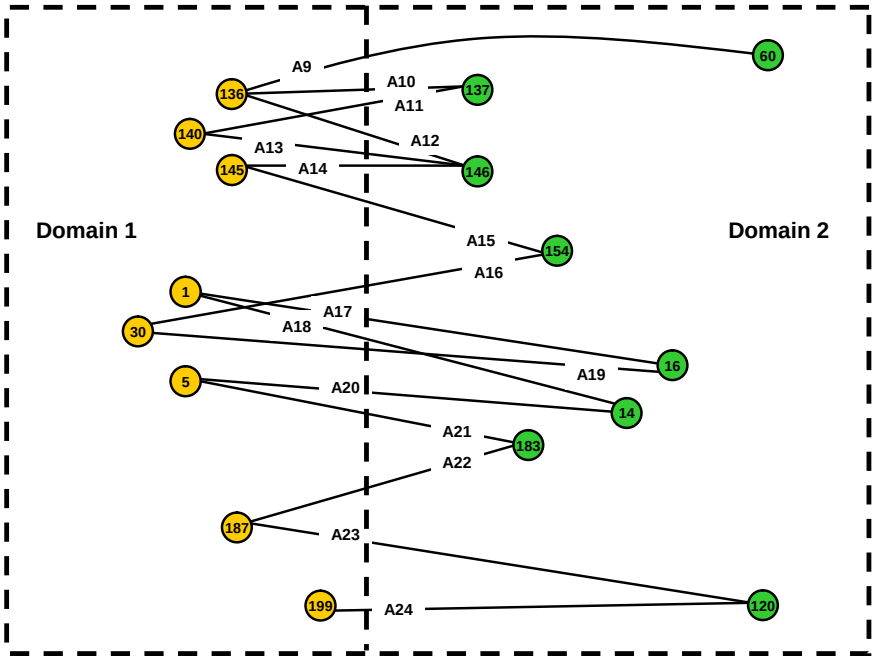


Figure 4.16. Peer-Peer Domains Large Services View.

represents the peering links between the domains (A_1 and A_2 in Figure 4.11). For each such link, one domain owns the link, and this domain models it as an observation node;the other domain models it as an SRG node. The second type of shared attribute describes whether a pair of servers can connect with each other. For each web service, the domain hosting the service models the shared attribute as an observation node while the domain on the client side models it as an SRG. In the evaluated peer-peer topologies, both domains observe the state of an event modeled by a shared attribute.

The construction of causal graphs for the peer-peer domain setting proceed similarly as with the provider-customer setting.

D. ABILENE-BASED TOPOLOGY

To provide an empirical evaluation using an established network topology, a topology was constructed based on the Abilene network backbone [1] (Figure 4.17). Customer domain connections to the Abilene network are modeled as stub routers. A

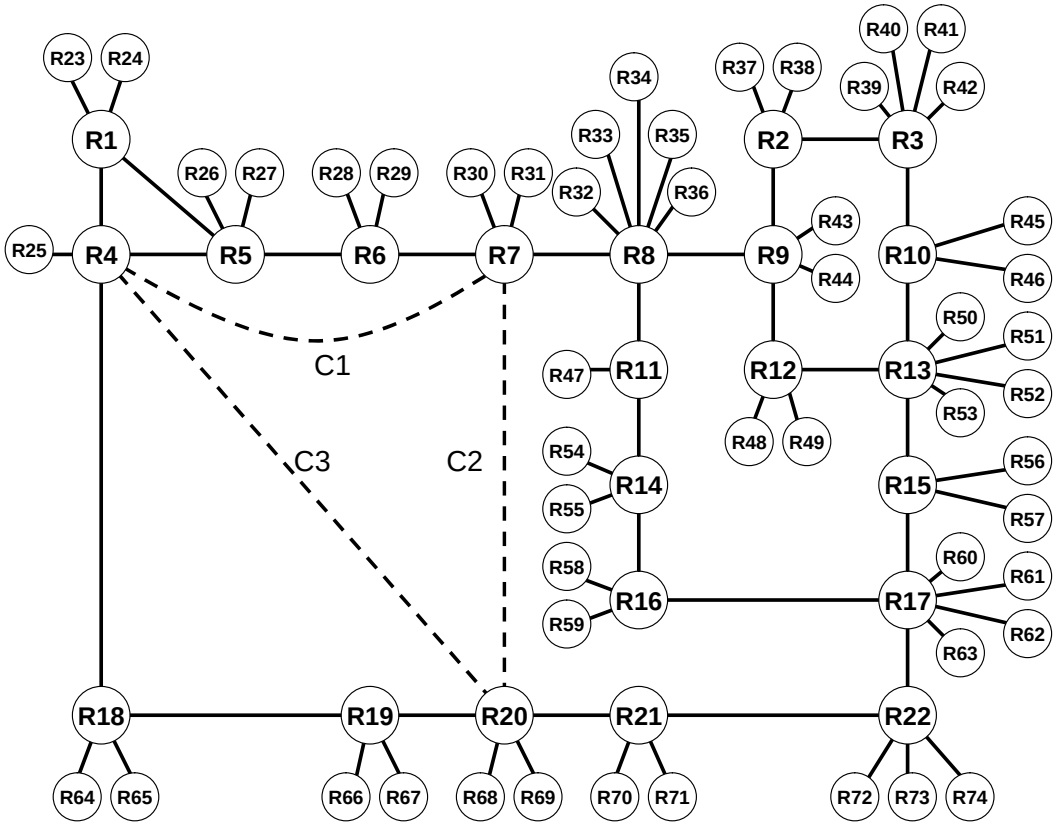


Figure 4.17. Abilene-based topology.

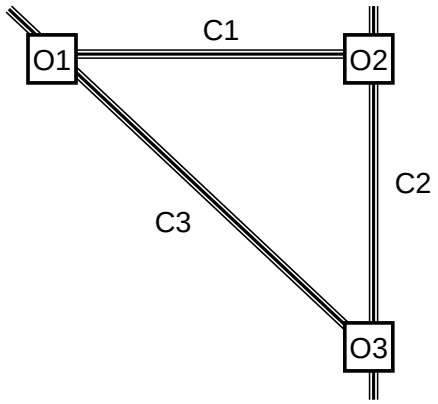


Figure 4.18. Provider domain to the Abilene network.

notional second network (Figure 4.18) was added across the Midwest United States to decrease the diameter of the Abilene-based network. This second network serves in the role as a provider to the Abilene network. The domain relationship and causal graph construction for the Abilene-based network are the same as for the provider-customer relationship described in Section B above, with the Abilene-based network filling the role of the customer. The Abilene-based network transits three circuits, $C1 \ldots C3$, provided by the added network. The gateway routers that connect to the circuits are routers $R4$, $R7$, and $R20$ in Figure 4.17.

E. INFERENCE ALGORITHMS

Two recent fault localization algorithms, SHRINK [19] and SCORE [23], were selected (as discussed in Chapter II), to evaluate the graph digest design. The use of two different algorithms is intended to provide evidence for the generality of the approach. Although neither algorithm was crafted to perform cross-domain fault localization, graph digests can be passed across domain boundaries for inclusion in a consolidated inference effort. The two algorithms are fundamentally different, with SHRINK using a Bayesian approach and SCORE using a greedy minimum set cover approach.

Performing isolated inference with SHRINK and SCORE is straightforward. Each domain performed inference, without benefit of collaboration, on their own causal graph. The resulting best explanations, B_1 and B_2 from domains 1 and 2 respectively, were then combined into $B_1 \cup B_2$. To implement full disclosure, a global causal graph for the two domains was created using a full view of the topologies, and then inference was performed on this global causal graph to derive B_u. Finally, for the graph digest approach, the causal graph of Domain 2 was first processed with the digest creation algorithm. The resulting digest was then combined with the causal graph of Domain 1 using the techniques described in Section 2, and inference was performed on the combined graph to produce B_d.

F. EVALUATION METRICS

1. Evaluating Accuracy

To evaluate the accuracy A and cost C metrics (defined in Chapter III), for each failure scenario, first the inference accuracy of isolated inference and full disclosure relative to ground truth was computed for each failure scenario. These results were compared to the accuracy achieved with the graph digest approach for the same failure scenarios. The digest creation algorithm presented in Figure 4.1 was used to create the graph digest for Domain 2. The equations 3.8 and 3.9 were applied to determine the accuracy A and cost C for each failure scenario.

Hypothesis testing showed whether the graph digest approach achieved better accuracy than isolated inference for each topology and inference method used. There are two domain relationships with three topology sizes each and two inference algorithms, for a total of twelve data sets using the synthetic topologies with which to test the hypothesis. Both SHRINK and SCORE performed inference on the Abilene-based topology, providing two additional data sets for hypothesis testing. With expected non-normal distributions, the hypothesis was evaluated using the Wilcoxon Signed-Rank Test at the 95% confidence level [12]. Specifically, the H_0 and H_1 hypotheses are that the graph digest approach achieves the same accuracy as isolated inference, and that the graph digest approach achieves better accuracy than isolated inference, respectively.

2. Evaluating Privacy Protection

The privacy protection provided by the prototype digest algorithm presented in Figure 4.1 was used. As discussed in Chapter III the practical approach was used to gauge the risk to privacy. Both of the evaluation inference algorithms, SHRINK and SCORE, were designed to find faults at the physical layer based on observations at the link and network layers. For this reason, sensitive properties that could be measured from such a reconstructed topology were selected.

In the evaluation, the following four sensitive properties were directly measured:

1. diameter - the diameter of the network.
2. number of routers - the total number of routers.
3. maximum node degree - the degree of the node with the highest degree.
4. reachability - whether or not an internal path can be inferred between two gateway routers.

For each of these sensitive properties, the ground truth values were the internal, or "hidden" values. As an example, an 18 router network with 3 visible gateways has 15 hidden routers. A digest revealing 3 internal routers reveals 3 of the 15 hidden routers. The diameter and degree sensitive properties are evaluated similarly, and the reachability property is binary.

Some may argue for treating IP link and web service failures themselves as a sensitive properties. Hiding evidence of failure, however, is both counter-productive, and directly opposed to the stated objective of sharing external observations of failure. Consider the provider-customer domain relationship as discussed above. The customer may need to give the provider specific information about failures so that the provider, who is contractually obligated to provide and maintain connectivity, to restore service. Similarly peer-peer domains may have contractual obligations to their respective customers.

No additional measures to hide sensitive properties were taken, but rather the digest creation algorithm (Figure 4.1) was evaluated for the inherent privacy protection provided. Removing sensitive information from a causal graph prior to digesting may provide additional protection, but the accuracy trade-offs must be understood. As shown in the final decision step in Figure 3.1, a digest is evaluated for compliance with a security policy after digesting and before dissemination. Ultimately, provable techniques are needed to evaluate the level of privacy disclosure for a digest. This

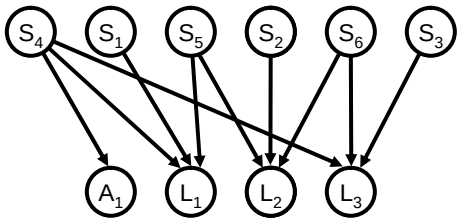


Figure 4.19. Example causal graph.

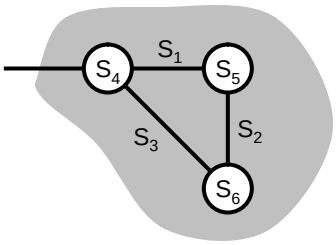


Figure 4.20. Topology produced by the attack heuristic.

research developed and implemented attack heuristics to learn sensitive properties of a network domain from its digested causal graph to gauge privacy protection.

The heuristic was implemented to specifically attack SHRINK-style bipartite causal graph digests. For brevity a simplified description of the attack heuristic is presented.

Consider the example causal graph in Figure 4.19. The causal graph has SRG nodes $S_1 \ldots S_6$, observation nodes $L_1 \ldots L_3$, and a shared attribute observation node A_1. Suppose it is known that A_1 represents a peering-point shared attribute. It can now be concluded that S_4 is a gateway router. Next observe that L_1 has 3 parent SRG nodes, with S_1 having a cardinality of 1. SRG S_1 is most likely a point-to-point link connecting the gateway S_4 and an adjacent router: S_5. Now nodes S_4 and S_5, and edge (S_4, S_5) are in a graph representing the topology. Applying the same reasoning with L_2 and L_3 allows adding node S_6 and edges (S_5, S_6) and (S_4, S_6), resulting in the topology shown in Figure 4.20.

The attack heuristic, which is presented at a high level in Figure 4.21, proceeds

DIGEST-ATTACK-HEURISTIC($Digest\ D = (SRG\ S, Observation\ O)$,
$SharedAttributes\ SA$)
1 EXTRACT-TOPOLOGY($Digest\ D, SharedAttributes\ SA$)
2 Add externally visible topology components
3 EVALUATE-PROPERTIES($Topology\ T = (Routers\ R, Links\ L)$)

Figure 4.21. Heuristic used to attack graph digests.

in three phases. First, the heuristic attempts to derive topology from a graph digest. Second, the heuristic adds missing externally visible components, such as gateway routers and transit links, to an extracted topology. Third, the heuristic uses the topology to estimate the values of sensitive properties. The attack heuristic (Figure 4.21) takes two parameters: the digest D and the shared attributes SA. Each shared attribute includes a distinction of whether or not it represents a peering point. The heuristic was designed to attack both an undigested bipartite causal graph, and a digest created using the prototype digest creation algorithm (Figure 4.1). Eventually, a collection of sample values is obtained for each property from each target set of scenarios.

A conservative approach to estimating the values was implemented, which represents an estimated lower bound on the information learned. In other words, the attack uncovers values for sensitive properties that are almost certainly true. As an example, suppose an attack uncovers two adjacent routers, each having an unresolved edge representing a connection to another router. The heuristic assumes that these unresolved edges connect to the *same* router.

Topology extraction, the first phase of the attack heuristic, is outlined in Figure 4.22. The procedure assumes a bipartite digest consisting of SRGs S, Observation nodes O, and edges E represented as adjacency lists. The procedure uses a function $adj()$ defined by

$$adj : O \cup S \rightarrow 2^O \cup 2^S,$$

where

$$adj(x) = \begin{cases} \{o \in O | (x, o) \in E\} & \text{if } x \in S \\ \{s \in S | (s, x) \in E\} & \text{if } x \in O \end{cases} \tag{4.1}$$

A discussion of topology extraction from a customer's causal graph in a provider-customer domain relationship (Figure 4.22) follows. The extraction begins by identifying candidate point to point links. The heuristic steps through all SRG nodes looking for any that points to just one observation node having three or fewer SRG parents. In a SHRINK-style model, a point-to-point link SRG will be a parent node of an IP link observation node between adjacent routers. Before any digest transformation and without any web services or VPN tunnels transiting the link, this observation node has three parent nodes: the point-to-point link and two routers. In this case, the point-to-point link SRG will have no other child observation nodes. However, the link may have one or more transiting VPN tunnels or web services, represented by a separate observation node. The point-to-point link will be a parent node of these observation nodes as well, each of which will likely have multiple (more than three) SRG parent nodes.

Now that some of the candidate point-to-point links have been identified, all down observation nodes are evaluated, starting with those having three or fewer parent SRGs. With the topologies used in this research, these comprise the majority of the observation nodes. Two points of clarification are:

1. Shared attribute nodes are never added as routers.
2. A tie breaker must be used when evaluating equal options.

An observation node having one parent SRG is most likely a router and link that have been aggregated, so that parent SRG node is added to the set of routers. An observation node with two parents likely represents a stub router and a link, so the node of highest degree is added to the set of routers. An observation node with three parents, the most common case in an undigested graph, most likely represents two

routers connected by a point-to-point link. The two parent nodes with the highest degree are added to the set of routers. If none of the observation node's parents are shared attributes, then a link between these routers is added to the topology.

If a down observation node has more than three parents, the heuristic assumes the observation node represents an IP tunnel or a web service. Often, any routers or links that can be learned will have already been captured by other observation nodes, which is why these observation nodes are processed last. The heuristic assesses each of these observation nodes to see if there are any non-shared attribute SRG parents not previously identified as routers or point-to-point links. If any such parents are found, up to two parents are added to the set of routers along with links between them if neither the observation node nor any of its parent SRGs is a shared attribute. Finally, the topology extraction heuristic looks at routers identified at the peering points and adds them as gateways.

The heuristic EXTRACT-TOPOLOGY evaluates a SHRINK-style bipartite causal graph representing network fault propagation, and extracts the network topology from the causal graph. The algorithm produces a graph of nodes representing routers, edges representing point-to-point links, and identifies IP tunnels. On an undigested graph the algorithm always returns an isomorphism of the original topology as shown below.

Without loss of generality, consider the customer topology extraction heuristic in Figure 4.24, with the following assumptions about the network topology graph. The graph is a simple, connected graph with at least two routers [1]. Shared attributes, gateway labels, and IP tunnels are not contained in the graph.

The following assumptions about the causal graph construction from the network topology are made. First, the causal graph is correctly constructed from a

[1]The requirement to have more than one router in a network graph is consistent with reverse-engineering a digest created using the digest creation algorithm (Figure 4.1). An isolated router influences no IP links, hence the algorithm would prune an SRG representing an isolated router from a causal graph.

connected network topology graph. Second, the SRG nodes are routers and point-to-point links, and the observation nodes are IP links between adjacent routers. Each pair of adjacent routers has exactly one IP link observation node modeled between them.

Let a network topology as described above be the graph $G = (V, E)$, where V consists of a set of two or more connected routers and E consists of a set of point-to-point links connecting the routers in V. Let the bipartite causal graph be $D = (S, O)$, where the SRG set S models all routers and point-to-point links in G, and the observation set O models observations of all IP links between each pair of adjacent routers in V. Let the network graph $G' = (V', E')$ be the topology constructed from reverse-engineering D. The heuristic EXTRACT-TOPOLOGY (Figure 4.22) iterates through observation nodes $o \in O$ to add edges $e' \in E'$ and previously undiscovered routers $v' \in V'$ to G'. The function $adj()$ is used as defined in Equation 4.1. To prove the correctness of EXTRACT-TOPOLOGY to reverse-engineer an isomorphism of G, namely G' this subsection will prove by induction that each step in adding the routers and edges to G' is correct.

The following axiom related to construction of the dependency graph D must be true for the Heuristic in Figure 4.22 to return an isomorphism G' of the original topology G from D.

Axiom 1. (Bipartite Causal Graph Construction).
An IP link directly connecting two routers is modeled as an observation node and depends on exactly three SRG parent nodes modeling two routers and the point-to-point link between the routers. The SRG modeling the point-to-point link affects no other observation nodes. An SRG modeling a router may affect many observation nodes, but a stub router affects only one.

Lemma 1. Provided that the above assumptions and Axiom 1 are true, an observation node $o \in O$ represents an IP link between two adjacent routers iff $|adj(o)| = 3$.

Proof. By Axiom 1, an observation node representing a point-to-point IP link between two adjacent routers is modeled as an observation node dependent on 3 SRG parent nodes.

Suppose an observation node $o \in O$ exists such that $|adj(o)| \neq 3$. Then by Axiom 1, o does not depend on precisely 3 SRG nodes, and hence o does not represent an IP link. □

Lemma 2. Provided that the above assumptions and Axiom 1 are true, an SRG node $s \in S$ represents either a point-to-point link or a stub router iff $|adj(s)| = 1$.

Proof. By Axiom 1, an SRG modeled as point-to-point link or a stub router affect exactly one observation node modeling an IP link.

Suppose an SRG s models a point-to-point link or stub router such that $|adj(s)| \neq 1$. There are two cases.

Case 1: $|adj(s)| = 0$. SRG s affects no IP links, hence s is an isolated component violating the assumptions that G contains more than one router in a connected simple graph.

Case 2: $|adj(s)| > 1$. SRG s affects more than one IP link, hence either s is not a point-to-point link or stub router, or G is not a simple graph. □

Lemma 3. Provided that the above assumptions and Axiom 1 are true, an SRG node $s \in S$ represents a router that is not a stub router iff $|adj(s)| \geq 2$.

Proof. By Axiom 1, an SRG modeled as a non-stub router affects exactly more than one observation node modeling an IP link.

Suppose SRG s models a router that is not a stub router and $|adj(s)| < 2$. There are two cases.

Case 1: $|adj(s)| = 0$. SRG s affects no IP links, hence s is an isolated component violating the assumptions that G contains more than one router in a connected simple graph.

Case 2: $|adj(s)| = 1$. SRG s affects exactly one IP link, hence by Lemma 2 either s is either a point-to-point link or stub router. □

Many lines in the heuristic (Figure 4.22) are not executed when processing an undigested graph using the above assumptions and axiom. The following two lemmas establish that many lines in the heuristic will not execute on such a graph, and thus can be removed to increase clarity.

Lemma 4. Provided that the above assumptions and Axiom 1 are true, all observation nodes $o \in O$ have degree $|adj(o)| = 3$.

Proof. By the assumptions only IP links are modeled in the graph, and by Lemma 1, IP links have degree $|adj(o) = 3|$.

Suppose an observation node $o \in O$ does not have a degree of 3. Then by Lemma 1, o does not model an IP link. By the above assumptions observation nodes only model IP links, hence o is not in the model. □

Lemma 5. Provided that the above assumptions and Axiom 1 are true, the algorithm correctly populates the temporary set of candidate point-to-point links C with point-to-point links and stub routers.

Proof. Lines 3-5 populate C with precisely the set of point-to-point links and stub routers by Lemma 2.

Suppose a point-to-point link or stub router SRG $s \in S$ is not placed in set C. Then there does not exist a single $o \in adj(s)$ such that $|adj(o)| = 3$, and by Lemma 2, s is neither a point-to-point link nor a stub router. □

The heuristic EXTRACT-TOPOLOGY (Figure 4.22) can now be simplified. The simplification is not necessary, but by removing lines that are not executed on a causal graph constructed from the network topology graph described above, a streamlined version of the algorithm can be used for the proof. First, by the above assumptions G is an undigested graph, so references to L_{up} inserted by the digest creation algorithm, can be removed from lines 4, 6, and 26. The loop in lines 35-37 can similarly be removed. By Lemma 4, lines 7-11 can be removed since an observation node will not have a degree of 1 or 2. By Lemma 5, all point-to-point links are added to container C, so lines 17 and 18 can be removed. By the assumptions, the graph has neither IP tunnels nor gateway labels, so lines 21-34 can be removed. Similarly, the tunnel set T can be removed from line 1. Since shared attributes are not considered, the set SA can be replaced by $\emptyset$. Line 3 can now be rewritten and lines 13 and 14 removed since the conditional **if**$(adj(o) \cap \emptyset) \neq \emptyset$ always evaluates to FALSE. The resulting simplified version of the algorithm is depicted in Figure 4.24.

Using the simplified version of the algorithm constructed from the heuristic in Figure 4.24, now the above axiom and lemmas will be applied to prove by induction on the number of IP links that EXTRACT-TOPOLOGY creates a G' that is isomorphic to an arbitrary G using D.

Theorem 1. Provided that the above assumptions and Axiom 1 are true, EXTRACT-TOPOLOGY correctly reverse-engineers an isomorphism G' of an arbitrary graph G with k IP links, from a bipartite causal graph D that models dependencies in G.

Proof. Base Case. Consider an arbitrary graph with $k = 1$ IP links, consisting of two routers and a point-to-point link (Figure 4.25 (a)). By Axiom 1, the IP link will be modeled as $o \in O$, dependent on two routers and a point-to-point link, each modeled as an $s \in S$. The loop in lines 3-5 will add the point-to-point link and both routers to set C. Since $|adj(o)| = 3$ by Lemma 1 and set $P = \emptyset$, lines 7 and 8 evaluate to TRUE. Since all three SRGs are in set C, set $R = \emptyset$, and $|adj(s)| = 1$ for each $s \in S$. A tie-breaker adds one of the SRGs $s \in S$ to P. Next, the other two SRGs $s \in S$ are added to the router set R in line 10. Finally, in step 11 an edge between the two routers in R is added to L. The algorithm completes with $V' = R$ and $E' = L$. The resulting graph G' contains two routers with a point-to-point link between them.

Inductive Step. Assume that an arbitrary graph with k IP links is always correctly reverse engineered, such that G' is isomorphic to G. It must be shown that EXTRACT-TOPOLOGY creates a G' isomorphic to an arbitrary graph G with $k + 1$ IP links.

Suppose a bipartite dependency graph D has been created from an arbitrary graph G with $k + 1$ IP links using Axiom 1. Consider temporarily removing the $(k + 1)^{th}$ observation node $o \in O$, all SRG parents $s \in adj(o)$ such that $|adj(s)| = 1$, and all edges into o. Assume that the inductive hypothesis is true, and that the first k observation nodes have been correctly reverse engineered to create an isomorphic graph $G'' \subset G$. By Lemma 1, the removed $(k + 1)^{th}$ observation node depends on a point-to-point link and two routers. By Lemma 2 SRG $s \in S$ models a point-to-point link or stub routers iff $|adj(s)| = 1$, hence these SRGs have been temporarily removed. Clearly G'' is missing a link and possibly a stub router that is in G. To prove that EXTRACT-TOPOLOGY creates a G' isomorphic to G, the $k + 1^{st}$ observation node and removed point-to-point link SRG, and any stub router SRG, must now be added to D. The $(k + 1)^{th}$ observation node must now be reverse engineered and added to G'', creating G'. There are three cases as shown in Figure 4.25 (b)...(d). The missing components in G'' after removing the $(k+1)^{th}$ IP link o_{k+1} are illustrated with dashed lines. In each case by Lemma 1 $|adj(o)| = 3$ and by Lemma 2 the SRG $s \in S$ modeling the point-to-point link in $adj(o)$ has degree of 1. The algorithm will add the SRG $s \in S$ representing the point-to-point link, along with any stub routers to temporary set C in lines 3-5. Lines 7 and 8 always evaluate to TRUE since $|adj(o)| = 3$ by Lemma 1, and $(adj(o) \cap P) = \emptyset$ since G is a simple graph. The three cases follow.

Case 1: A point-to-point link and stub router are in G, but missing from G'' (Figure 4.25 (b)). Either an SRG modeling the missing stub router or the point-to-point link in $adj(o)$ is added to P in line 9. In line 10, the other two SRGs are added

to R. One router, $s \in S$ has been previously reverse engineered, but since $s \in R$, $R \cup \{s\} = R$. In line 11, a link is added between the two routers. Now that the components in G missing from G'' have been added, G' is isomorphic to G.

Case 2: A link in G is missing from G'', and G'' consists of two disconnected components (Figure 4.25 (c)). Let $o \in O$ be the $k+1^{st}$ observation node. Since both routers $s \in adj(o)$ have been reverse-engineered by the first k observation nodes, $|adj(s)| > 1$ for both SRG nodes representing the routers. Thus, $|C| = 1$, and C contains just the point-to-point link. The link is added to P in line 9 and both routers are re-added to R in line 10. Since $s \in R$ for each router, $R \cup \{s\} = R$. In line 11, the link is added between the two routers. Now that the link in G missing from G'' has been added, G' is isomorphic to G.

Case 3: A link in G is missing from G'', and G'' consists of one connected component (Figure 4.25 (d)). Let $o \in O$ be the $k+1^{st}$ observation node. Since both routers $s \in adj(o)$ have been reverse-engineered by the first k observation nodes, $|adj(s)| > 1$ for both SRG nodes representing the routers. Thus, $|C| = 1$, and C contains just the point-to-point link. The link is added to P in line 9 and both routers are re-added to R in line 10. Since $s \in R$ for each router, $R \cup \{s\} = R$. In line 11, the link is added between the two routers. Now that the link in G missing from G'' has been added, G' is isomorphic to G.

□

Any externally visible components that are missing from the extracted topology are then added. The topology is then fed to the next heuristic, which is property evaluation.

Topology extraction for the peer-peer domain relationship (Figure 4.23) proceeds similarly to extraction for the provider-customer relationship with the following three modifications. First, the phase to identify components based on point-to-point links first processes all non-SA observation nodes, then all SA observation nodes. Second, for the case of an SA observation node with two SRG parent nodes, the heuristic adds only one router. Third, the heuristic processes observation nodes as well as SRG nodes to identify gateway routers, and creates a peering link instead of a transit link.

The property evaluation phase shown in Figure 4.26, measures the four sensitive properties previously identified: reachability, domain diameter, number of routers, and degree of the node with the highest degree. This heuristic takes the

topology generated by the heuristic in Figure 4.22 and a pair of gateway routers for reachability testing as input parameters.

The heuristic first iterates through the routers in the topology to find the node with the maximum degree. Next, the heuristic temporarily removes all peering links from the topology. If the reachability nodes have been identified in the topology as gateways and a path exists between them, internal reachability is determined to be true.

To calculate the number of routers, the heuristic first needs to resolve unresolved edges. A router has an unresolved edge in the topology if the router has an edge to L_{up} in the causal graph. The heuristic attempts to resolve these unresolved edges by checking to see if a link between routers with unresolved edges can be established. Any number of routers may resolve their edges with a single router having an unresolved edge. This detail is consistent with the logical OR implementation in the digest algorithm (Figure 4.1), since an edge to L_{up} represents one or more connections.

The heuristic adds all routers with an unresolved edge to set U, and colors each router *white*. For each pair of routers in set U, if the routers are not adjacent and both routers are not gateways, then the routers are colored *black* and an edge is added between the routers. After adding these edges, which are a necessary step to find the lower bound on the network diameter, the heuristic next checks to see if there is a gateway router that is colored *white*. Since external information is visible with respect to the domain attacking a digest and it is not common practice to peer with multiple domains from a single gateway router, the heuristic explains the unresolved edge by adding an internal router to the topology. If instead no gateway routers are colored *white* but one or more internal routers are colored *white*, the heuristic explains the first instance as an external connection to an unknown domain, and a subsequent instance as an internal router. Dual homing is common practice for network domains, but dual homing with an arbitrary number of connections is not.

If an internal router is added to a domain topology, every router in set U will be able to resolve its unresolved edge with the added router.

Finally, the heuristic uses the Floyd-Warshal algorithm [10] to determine the network diameter. The Floyd-Warshal algorithm computes the longest of the shortest paths in a graph.

3. Evaluating Scalability

The first ten double failure scenarios from each topology were instrumented to measure the real elapsed time using the Linux time command via Cygwin. To evaluate the scalability metric E the inference time was measured using both the full disclosure and graph digest approaches. The mean of each set of ten measurements was used as input to equation (3.11). The simulations were run on a 1.61 GHz computer with 960 MB RAM running Windows XP, service pack 3.

G. CONCLUSION

This chapter outlines the test methodology used to evaluate the hypothesis that an approach using the framework in Chapter III achieves better accuracy than isolated inference. Six different network domain topologies built from realistic network topology atoms were used with varying sizes and two different domain peering relationships. An additional topology based on the Abilene [1] backbone infrastructure was used. The prototype graph digest algorithm (Figure 4.1) was evaluated using two different intra-domain fault localization algorithms: SHRINK and SCORE. The variety in topologies, as well as the use of two different inference algorithms, serves to evaluate the generality of the framework described in Chapter III. By measuring performance in terms of the accuracy, privacy, and scalability metrics outlined in Chapter III, the evaluation results in Chapter V establish the feasibility of the graph digest approach for cross-domain fault localization.

```
EXTRACT-TOPOLOGY(Digest D = (SRG S, Observation O), SharedAttribute SA)
 1  initialize router set R, link set L, and tunnel set T to ∅
 2  initialize temporary set variables C and P to ∅
    ▷ Identify candidate point-to-point links
 3  for each SRG s ∈ (S − SA)
 4        do if there exists a single o ∈ adj(s) such that o ≠ L_up and |adj(o)| ≤ 3
 5              then add s to C
    ▷ Identify components based on point-to-point links
 6  for each observation node o ≠ L_up in O
 7        do if |adj(o)| = 1
 8              then add s ∈ (adj(o) ∩ (S − SA)) to R
 9           elseif |adj(o)| = 2
10              then add each s ∈ (adj(o) − (SA ∪ P)) to R
11                   add edge (s_i, s_{j≠i}) for each s_i, s_{j≠i} ∈ (adj(o) ∩ R) to L
12           elseif |adj(o)| = 3
13              then if (adj(o) ∩ SA) ≠ ∅
14                      then add each s ∈ (adj(o) − (SA ∪ P) to R
15                      else  if (adj(o) ∩ P) = ∅
16                               then add s ∈ ((adj(o) ∩ C) − R) with min. |adj(s)| to P
17                                    if (adj(o) ∩ P) = ∅
18                                       then add s ∈ (adj(o) − R) with min. |adj(s)| to P
19                                    add each s ∈ (adj(o) − P) to R
20                            add edge between each s ∈ (adj(o) ∩ R) to L
    ▷ Identify gateway routers based on shared attribute peering points
21  for each s ∈ (S ∩ SA) such that s is a peering point shared attribute
22        do for each o ∈ (O ∩ adj(s))
23              do if |adj(o) − SA| ≤ 2
24                    then label each s ∈ (adj(o) ∩ R) as gateway
25                         add edge between each s ∈ (adj(o) ∩ R) and label as transit
    ▷ Identify routers based on logical tunnels
26  for each observation node o ≠ L_up in O such that |adj(o)| > 3
27        do initialize a new tunnel t to ∅
28           if |adj(o) ∩ R| < 2
29              then if |adj(o) ∩ R| = 0
30                      then add s ∈ (adj(o) − (P ∪ SA)) with max. |adj(o)| to R
31                   add s ∈ (adj(o) − (R ∪ P ∪ SA)) with max. |adj(o)| to R
32           for each s ∈ adj(o) ∩ R
33                do add router s to tunnel t
34           T ← T ∪ {t}
    ▷ Identify unresolved edges
35  for each s ∈ adj(L_up)
36        do if s ∈ R
37              then mark s with unresolved_edge ← TRUE
```

Figure 4.22. Heuristic to extract router, link, and tunnel sets of customer topology from a graph digest.

```
EXTRACT-TOPOLOGY(Digest D = (SRG S, Observation O), SharedAttribute SA)
 1  initialize router set R, link set L, and tunnel set T to ∅
 2  initialize temporary set variables C and P to ∅
    ▷ Identify candidate point-to-point links
 3  for each SRG s ∈ (S − SA)
 4        do if there exists a single o ∈ adj(s) such that o ≠ L_up and |adj(o)| ≤ 3
 5              then add s to C
    ▷ Identify components based on point-to-point links
    ▷ Do for all non-SA observation nodes first, then for all SA observation nodes
 6  for each observation node o ≠ L_up in O
 7        do if |adj(o)| = 1
 8              then add s ∈ (adj(o) ∩ (S − SA)) to R
 9           elseif |adj(o)| = 2
10              then if o ∈ (O ∩ SA) such that o is a peering point shared attribute
11                        then add s ∈ (adj(o) − (SA ∪ P)) with max. |adj(o)| to R
12              else  add each s ∈ (adj(o) − (SA ∪ P)) to R
13                    add edge to L between each s ∈ (adj(o) ∩ R)
14           elseif |adj(o)| = 3
15              then if (adj(o) ∩ SA) ≠ ∅
16                        then add each s ∈ (adj(o) − (SA ∪ P) to R
17                        else  if (adj(o) ∩ P) = ∅
18                                 then add s ∈ ((adj(o) ∩ C) − R) with min. |adj(s)| to P
19                                       if (adj(o) ∩ P) = ∅
20                                          then add s ∈ (adj(o) − R) with min. |adj(s)| to P
21                              add each s ∈ (adj(o) − P) to R
22                              add edge between each s ∈ (adj(o) ∩ R) to L
    ▷ Identify gateway routers based on shared attribute peering points
23  for each s ∈ (S ∩ SA) such that s is a peering point shared attribute
24        do for each o ∈ (O ∩ adj(s))
25              do if |adj(o) − SA | ≤ 2
26                        then label each s ∈ (adj(o) ∩ R) as gateway
27                             add edge between each s ∈ (adj(o) ∩ R) and label as peering
28  for each o ∈ (O ∩ SA) such that o is a peering point shared attribute
29        do if |adj(o)| ≤ 2 and |adj(o) ∩ R| = 1
30              then label s ∈ (adj(o) ∩ R) as gateway
    ▷ Identify routers based on logical tunnels
31  for each observation node o ≠ L_up in O such that |adj(o)| > 3
32        do initialize a new tunnel t to ∅
33           if |adj(o) ∩ R| < 2
34              then if |adj(o) ∩ R| = 0
35                        then add s ∈ (adj(o) − (P ∪ SA)) with max. |adj(o)| to R
36                   add s ∈ (adj(o) − (R ∪ P ∪ SA)) with max. |adj(o)| to R
37           for each s ∈ adj(o) ∩ R
38                do add router s to tunnel t
39           T ← T ∪ {t}
    ▷ Identify unresolved edges
40  for each s ∈ adj(L_up)
41        do if s ∈ R
42              then mark s with unresolved_edge ← TRUE
```

Figure 4.23. Heuristic to extract router, link, and tunnel sets of a peering topology from a graph digest.

EXTRACT-TOPOLOGY($Digest\ D = (SRG\ S, Observation\ O), SharedAttribute\ SA$)

```
1  initialize router set R and link set L to ∅
2  initialize temporary set variables C and P to ∅
   ▷ Identify candidate point-to-point links
3  for each SRG s ∈ S
4        do if there exists a single o ∈ adj(s) such that |adj(o)| ≤ 3
5              then add s to C
   ▷ Identify components based on point-to-point links
6  for each observation node o ∈ O
7        do if |adj(o)| = 3
8              then if (adj(o) ∩ P) = ∅
9                        then add s ∈ ((adj(o) ∩ C) − R) with min. |adj(s)| to P
10                    add each s ∈ (adj(o) − P) to R
11           add edge between each s ∈ (adj(o) ∩ R) to L
```

Figure 4.24. Algorithm to extract router, link, and tunnel sets of customer topology from an undigested graph digest.

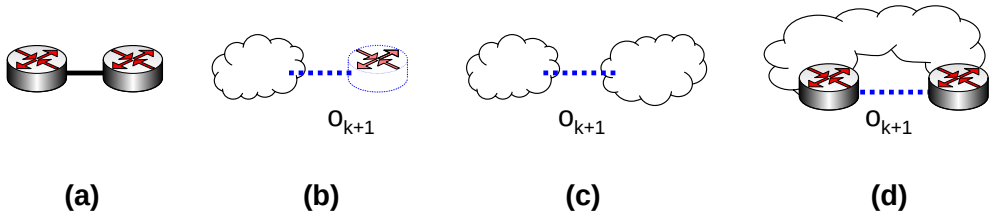


Figure 4.25. (a) Base Case and (b)...(d) possible cases for G''.

```
EVALUATE-PROPERTIES(Topology = (Routers R, Links L))
 1  reachability ← FALSE
 2  diameter ← 0
 3  number_of_routers ← 0
 4  max_node_degree ← 0
    ▷ Determine reachability
 5  Temporarily remove all peering links from L
 6  if reachability nodes r1 ∈ R and r2 ∈ R are labeled as gateway
 7     then if a path r1 ⇝ r2 exists
 8             then reachability ← TRUE
 9  Replace all peering links
    ▷ Find the maximum node degree
10  for each router r ∈ R
11        do if degree of r > max_node_degree
12              then max_node_degree ← degree of r
    ▷ Find the number of routers
13  Add all routers in R that have an unresolved edge to temporary set U
14  Color each router in U as white
15  external_connection_added ← FALSE
16  while exists a router in U colored white
17        do for all r ∈ U
18              do for all (s ≠ r) ∈ U such that s is not marked gateway
19                    do if edge (r, s) ∉ L
20                          then add (r, s) to L
21                               color r black
22                               color s black
23           if exists r ∈ U colored white and marked gateway
24             then add new router s to R
25                  add s to U
26                  color s white
27             else  if exists r ∈ U colored white and not marked gateway
28                     then if external_connection_added = FALSE
29                             then external_connection_added ← TRUE
30                                  color r black
31                             else add new router s to R
32                                  add s to U
33                                  color s white
34  number_of_routers ← |R|
    ▷ Find the network diameter
35  Run Floyd-Warshal algorithm on T to derive diameter
```

Figure 4.26. Heuristic to evaluate reachability, diameter, number of routers, and maximum node degree sensitive properties from a graph digest.

V. EVALUATION RESULTS

This chapter presents the evaluation results for the model presented in Chapter III. The evaluation methodology described in Chapter IV is used.

A. PROVIDER-CUSTOMER SETTING

1. Accuracy Evaluation Results - SHRINK

For all but 5 of 377 tested scenarios, $\alpha_d \geq \alpha_s$, resulting in non-negative accuracy improvement scores A (Figure 5.1). The average A score was 0.31, 0.36, and 0.42 for the small, medium, and large topology respectively. The maximum score for each topology was 1.0. There was an accuracy improvement in 55%, 66%, and 68% of the test scenarios for the small, medium, and large topology respectively (indicated by an oval in Figure 5.2). The results indicate that scaling the domain size has little impact on the accuracy of B_d, $B_1 \cup B_2$, or A with respect to B_T.

All instances of accuracy $A = 1.0$ in Figure 5.1 reflect failure scenarios for

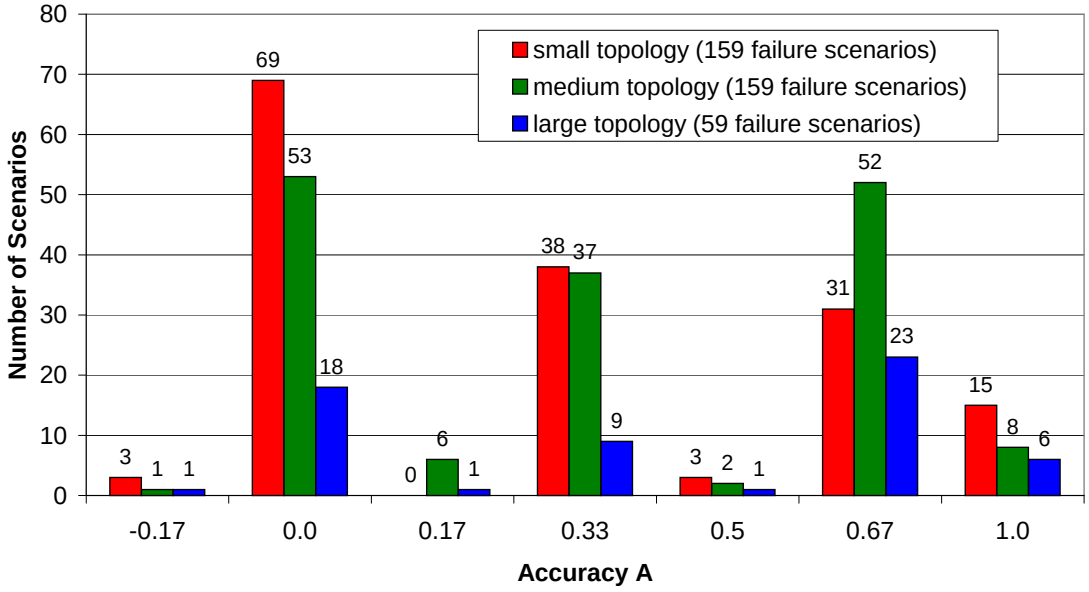


Figure 5.1. Histogram of A metric for the provider-customer setting.

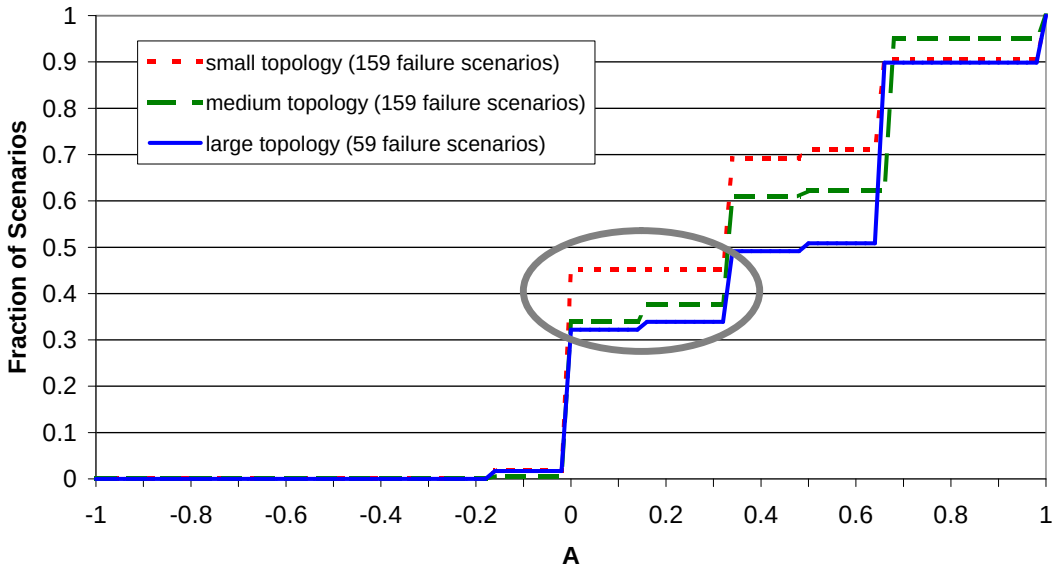


Figure 5.2. CDF of A metric for the provider-customer setting.

which only components in the provider (Domain 1) have failed. As discussed in Chapter IV, failures in the provider domain are not observable by the provider. While the customer can observe failures, inference by the customer in isolation will only result in either incorrectly identified components in the customer domain, or finger pointing at the provider. While it is not surprising that isolated inference was wholly unsuccessful in identifying the failures ($\alpha_s = 0.0$) for these scenarios, the graph digest approach did achieve perfect ($\alpha_d = 1.0$) accuracy.

The five negative accuracy results stem from double-failure scenarios of components in the same neighborhood of a router or switch. Four of the five failure scenarios with a negative score were essentially the same failure scenario that occurred four times. The failure scenarios in the small topology graph were $\{F_1, R_{11}\}$, $\{O_1, R_{11}\}$, and $\{F_2, R_{11}\}$. Considering that F_1, O_1, and F_2 are indistinguishable in the inference model, they were essentially the same failure scenario. Furthermore, the failure scenario in the medium topology $\{F_2, R_{11}\}$ was identical to the failure scenario by the same name in the small topology. For each of these four scenarios, $B_d = B_u = \{O_2, R_{11}\}$, resulting in a false positive (O_2), and a false negative

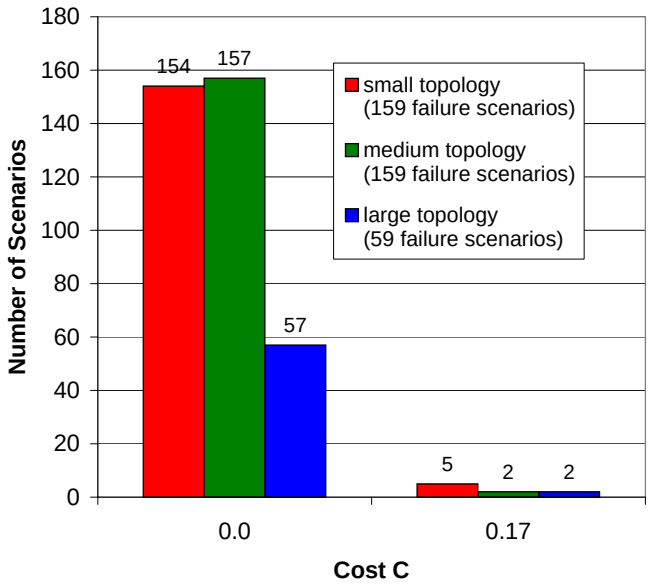


Figure 5.3. Histogram of C metric for the provider-customer setting.

($\{F_1, O_1, F_2\}$). Isolated inference ($\cup_i B_i$) hypothesized $\{R_{11}\}$ resulting in a single false negative result.

The negative score in the large topology occurred for the failure scenario $\{F_3, P_{10-11}\}$, where P_{10-11} is the point-to-point link between R_{10} and R_{11}. For this scenario $B_d = \{F_3, R_{11}\}$, with a false positive result (R_{11}) and a false negative result (P_{10-11}). $B_u = \{F_3\}$ and $\cup_i B_i = \{P_{10-11}\}$, each with a single false negative result.

The cost metric C depicted in Figures 5.3 and 5.4 shows a minimal cost in using the digest approach. The cost to inference equaled zero in all but nine test cases, meaning that the digest approach achieved the same accuracy as the full disclosure approach in 97.6% of the 377 tested scenarios. The average score was 0.005, 0.002, and 0.006 for the small, medium, and large topologies respectively, with a maximum value of 0.17 in each topology. Each of the nine failure scenarios for which $C > 0.0$ occurred when two components failed simultaneously in the same neighborhood of a router or switch. For each of these nine scenarios, the best explanation using the graph digest approach returned one correct and one incorrect failure while the full disclosure approach returned one correct failure and did not hypothesize a second

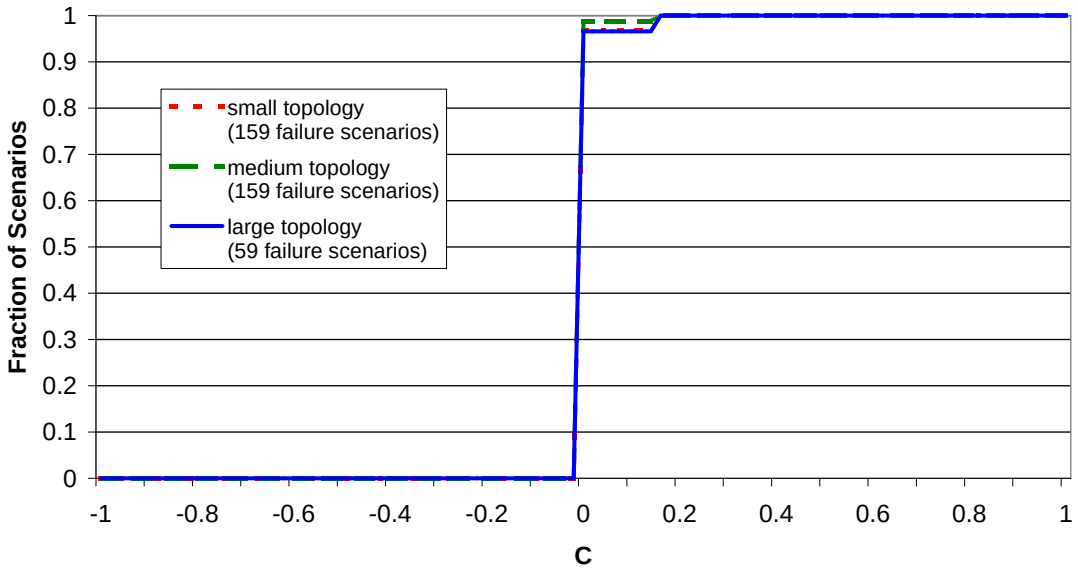


Figure 5.4. CDF of C metric for the provider-customer setting.

failure.

The digest algorithm in Figure 4.1 potentially degrades A. The logical-OR treatment for edges to L_{up} removes information about conditional dependencies on an SRG from the joint distribution, reducing the probability the SRG is up given the state of the remaining dependent observation nodes. Additionally, the aggregation step, exaggerated by using uniform prior probabilities, lumps additional SRGs into a best explanation for α_d. Since all equipment identified in a hypothesis would have to be checked, all SRGs that have been aggregated into an SRG are unraveled into a best explanation. Consequently, aggregation potentially adversely affects h_d, and ultimately α_d and A. In spite of the information loss, the graph digest approach performed remarkably well as discussed above.

The Z values using the Wilcoxon Signed-Rank Test are 8.21, 8.92, and 5.44 for the small, medium, and large topologies respectively. The hypothesis H_1 passed the 95% confidence test for SHRINK in the Provider-Customer setting.

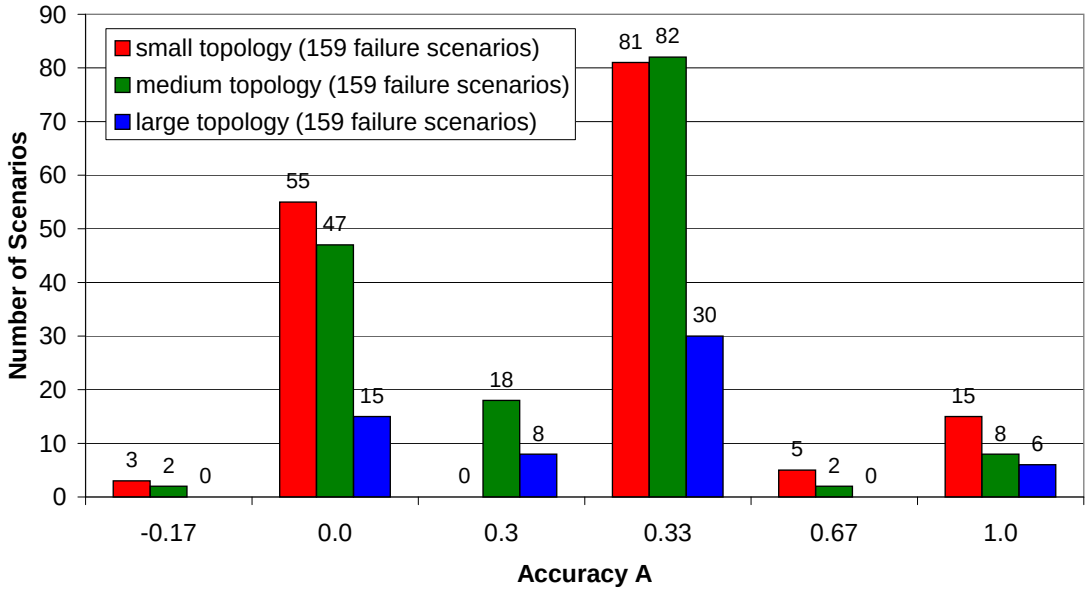


Figure 5.5. Histogram of A metric for the provider-customer setting.

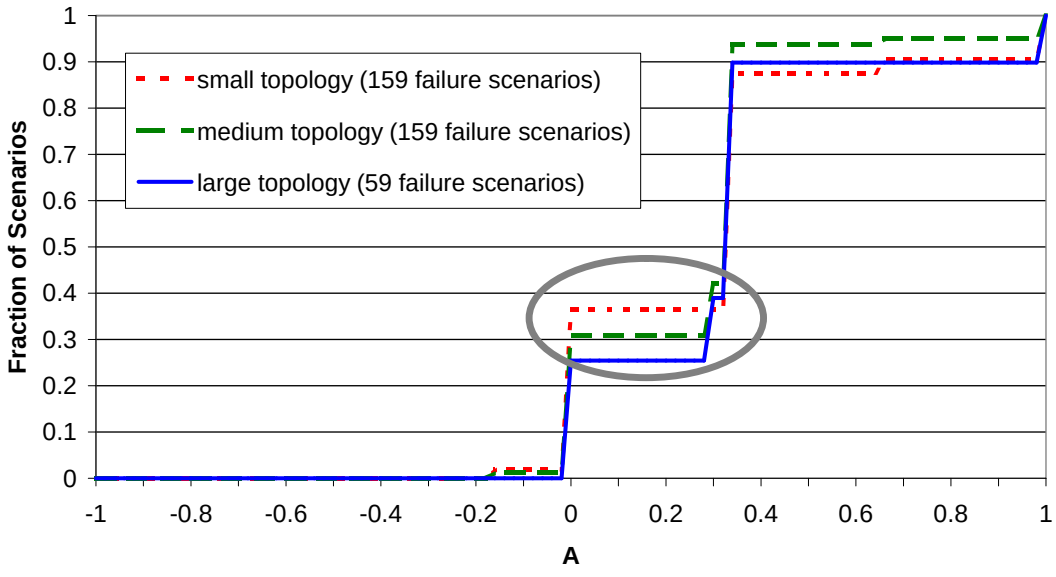


Figure 5.6. CDF of A metric for the provider-customer setting.

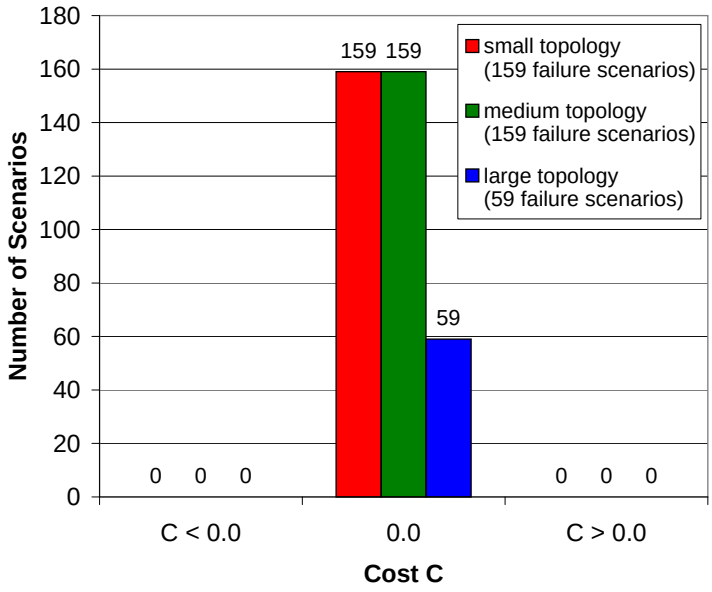


Figure 5.7. Histogram of C metric for the provider-customer setting.

2. Accuracy Evaluation Results - SCORE

For all but 5 of 377 tested scenarios $\alpha_d \geq \alpha_s$, resulting in non-negative accuracy improvement scores A (Figure 5.5). The average score is 0.28, 0.26, and 0.31 for the small, medium, and large topology respectively. The maximum score for each topology is 1.0. An accuracy improvement in 57%, 45%, 34% of the test scenarios was observed for small, medium, and large topology respectively (indicated by an oval in Figure 5.6). The results indicate that scaling the domain size has little impact on the accuracy of B_d, $B_1 \cup B_2$, or A with respect to B_T.

The five instances for which accuracy scores are negative occurred in double-failure scenarios. For each of these scenarios isolated inference returned only one identified failure, resulting in one correct explanation and one false positive result. The graph digest and full disclosure approaches returned one correct and one incorrect explanation, yielding both a false positive and a false negative in the hypothesis. As in SHRINK, each of the scenarios with a negative score occurred when two failures occurred in the same neighborhood in the physical topology graph.

The cost metric C, depicted in Figures 5.7 and 5.8, is uniform at 0.0 across all

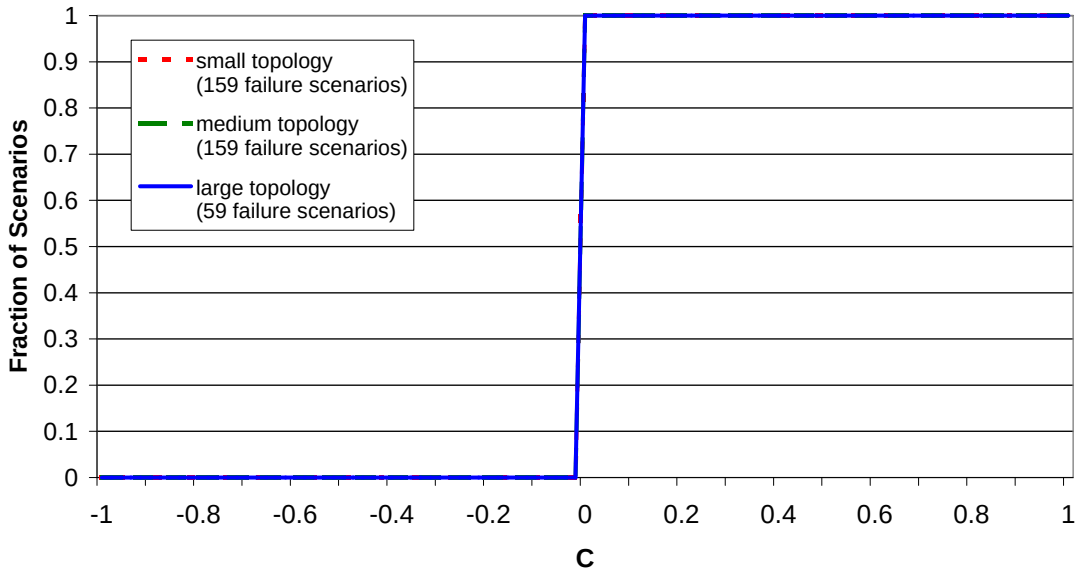


Figure 5.8. CDF of C metric for the provider-customer setting.

three topologies. This means that for all 377 tested scenarios, $B_d = B_u$. Interestingly, when isolated inference outperformed the graph digest approach for the five failure scenarios discussed above, isolated inference also outperformed full disclosure for these same scenarios.

The digest algorithm in Figure 4.1 potentially degrades A using SCORE. The impact of the digest's logical-OR step to SCORE is to artificially inflate the hit ratio of every SRG having more than one "up" observation set member. The aggregation step has no impact on SCORE as the SCORE algorithm performs this step during preprocessing. As with SHRINK, all SRGs that have been aggregated into an SRG are unraveled in a best explanation. In spite of the information loss, the graph digest approach performed remarkably well as discussed above.

The Z values using the Wilcoxon Signed-Rank Test were 8.83, 8.93, and 5.78 for the small, medium, and large topologies respectively. The hypothesis passed the 95% confidence test using SCORE in the Provider-Customer setting.

	Small	Medium	Large
Degree	2.11 (4)	3.28 (5)	4.33 (6)
Diameter	3.27 (4)	9.95 (11)	22.01 (23)
Routers	12.06 (15)	47.71 (51)	197.97 (201)

Table 5.1. Privacy metric rMSE versus (true value).

	Node Degree	Domain Diameter	Number Routers
gSTD	1.09	0.59	1.66
E(X)	2.01	0.94	3.16

Table 5.2. Privacy metric gSTD versus sample mean.

3. Privacy Evaluation Results

The digest creation algorithm (Figure 4.1) creates identical digests for both SHRINK and SCORE. As a consequence, the privacy results for SHRINK and SCORE are identical.

To compute the privacy protection for the customer, each digest was attacked using the heuristic described in Chapter IV. The attack heuristic adds any missing externally visible gateway routers and transit IP links to each topology extracted from a digest. As previously discussed, no attempts were made to hide information and no post-processing of the digests was performed to reduce the information leaked, but rather the design was tested to see how much information leaked using the simple digest algorithm described in Figure 4.1.

As depicted in Table 5.1, the root mean squared error (rMSE) was high relative to the true value for the sensitive properties. The outcome for privacy evaluation means that the information learned from the attacks was generally far from the true values. Table 5.2 shows that the generalized standard deviation (gSTD) for each privacy metric was low compared to the mean. This result means that there is little variation in the amount of information learned about each sensitive property from each digest attack. These results suggest a reasonable level of privacy protection

considering the use of a prototype digest creation algorithm for the attack heuristic.

To provide more detail, histograms of the attack estimates are shown. Additionally, the relative error of the attack results versus the true value for each sensitive property was calculated. The results expressed with histograms and cumulative distribution functions (CDF), less reachability, are presented for each domain size in Figures 5.9 - 5.14. The reachability results are discussed next.

The reachability metric is binary with a one, the true value, representing internal reachability between two externally visible gateways in the customer domain. The reachability test was conducted between gateway routers $R11$ and $R12$ in Figures 4.2, 4.5, and 4.6. The gateways are 2 hops apart, with router $R13$ connecting them.

Only 7 of the 377 evaluated failure scenarios identified the reachability between the nodes. Six of the failure scenarios that revealed reachability involved failure of router $R13$ with a second failure. The second failed component in each case caused a failure observation that *could* have been caused by failure of $R11$ or $R12$. For example, in the failure scenario $\{R13, F3\}$, the IP link between $R11$ and $R12$ is observed to be down. Since the failure of either $R11$ or $R12$ would cause this link to observe failure, this is a failure scenario that reveals reachability between the nodes. The seventh failure scenario to reveal the reachability was the failure scenario $\{R11, R12\}$.

The network diameter values measured by the attack are presented in Figure 5.9. The histogram clearly shows little variation in the attack estimates. At 50% mass of the experiments (Figure 5.10), the relative error was 75%, 91%, and 96% for the small, medium, and large topologies respectively. This result bodes well for the inherent protection provided by the digest approach as a network domain size scales up.

A histogram showing the number of routers found by the attack heuristic is presented in Figure 5.11. The histogram shows fairly consistent results for each topology size. As shown in Figure 5.12, the relative error between the number of routers in a network and the number detected from attacking a digest increased

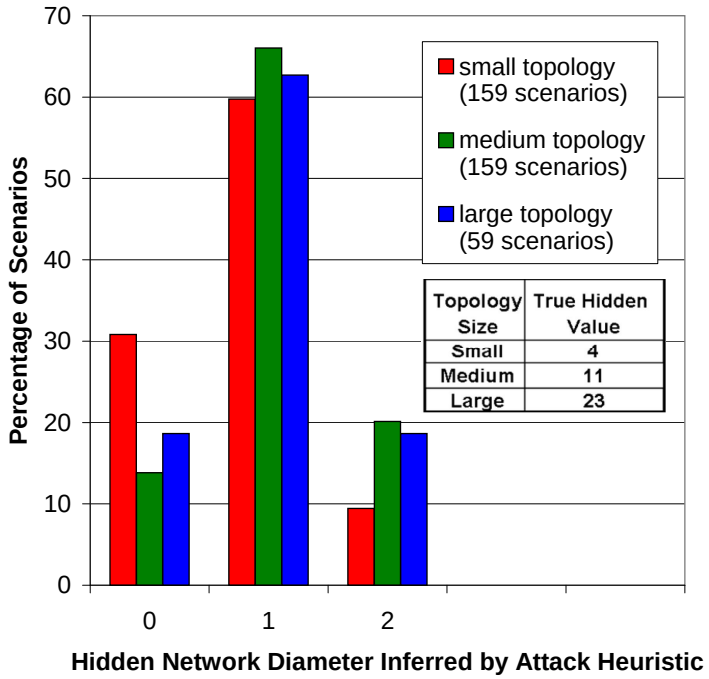


Figure 5.9. Histogram for the diameter property.

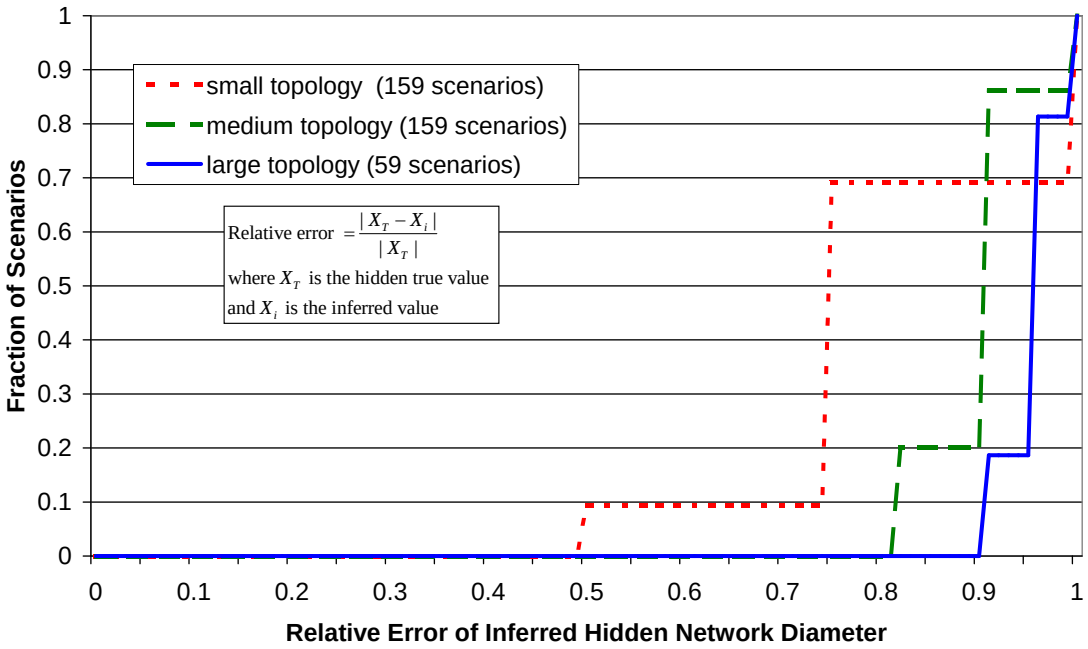


Figure 5.10. CDF relative error $|X_T|$ for the diameter property.

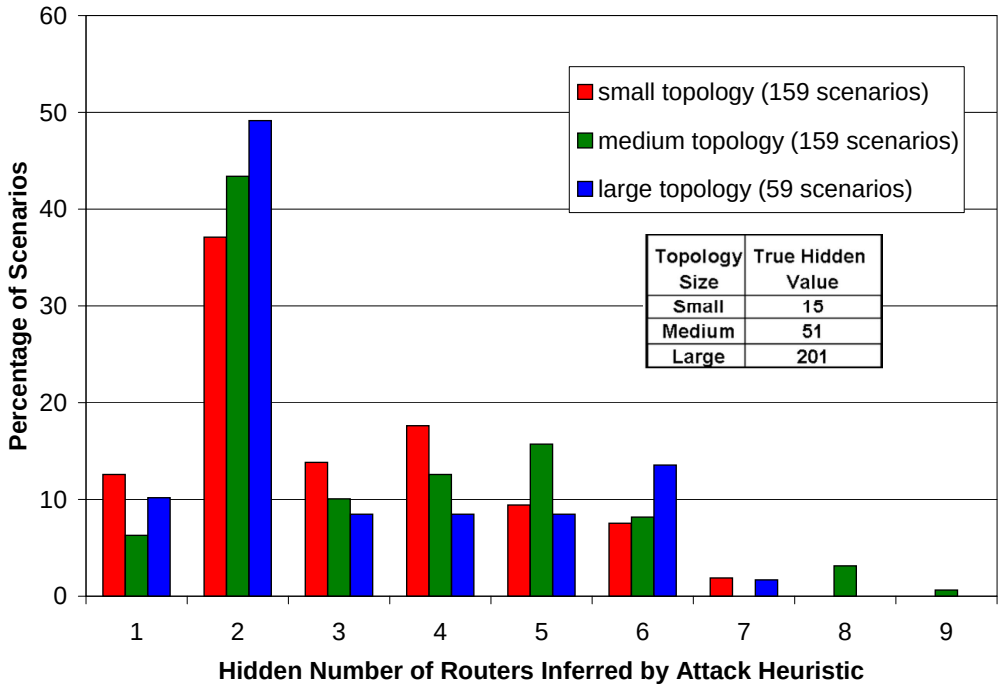


Figure 5.11. Histogram for the number of routers property.

with topology size. At 50% mass of the experiments (Figure 5.12), the relative error was 80%, 94%, and 99% for the small, medium, and large topologies respectively. Intuitively these results makes sense since each digest only provides a small collection of nodes. In general the topology learned consists of a neighborhood around one or two routers, and multiple failures whose neighborhoods intersect allow a larger portion of the topology to be inferred.

When a failure impacts an IP tunnel, as do 78% of the failure scenarios, information about the neighborhood around each router on the tunnel is potentially revealed. The IP tunnels do have an inherent protection feature in that an observation node representing the IP tunnel will most likely have more than three parents in a digest. This creates ambiguity in reconstructing router adjacencies along the tunnel for the attack heuristic used.

The topologies were seeded with an unfavorable setting for the node degree sensitive property by placing a router with high degree at the gateway in the customer

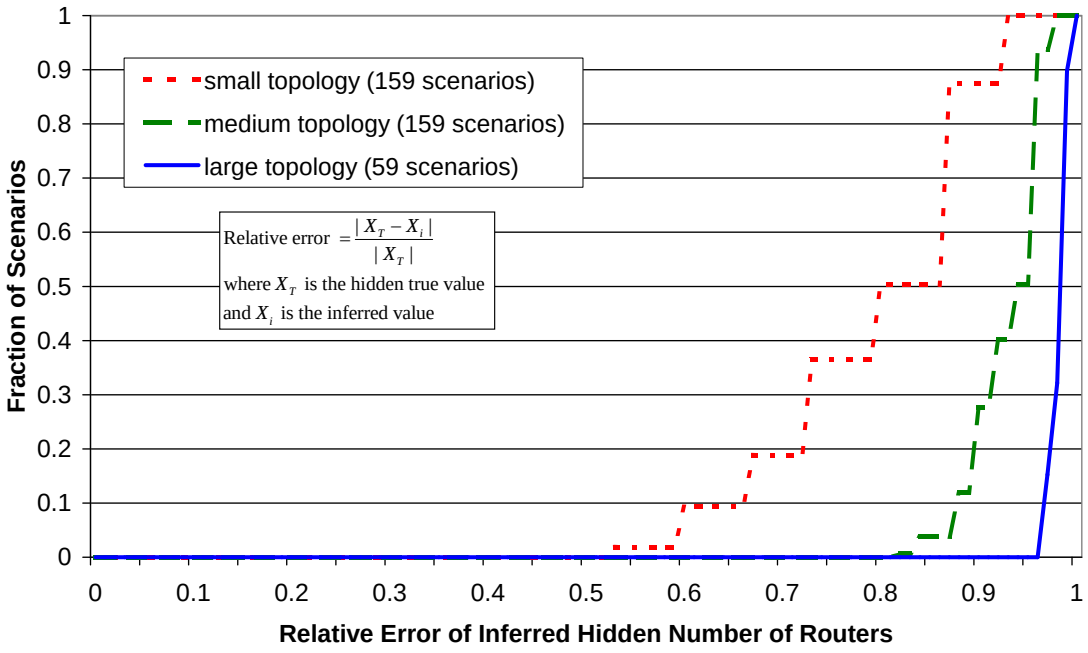


Figure 5.12. CDF relative error for the number of routers property.

domain. The node aggregation and Noisy-OR steps performed by the digest algorithm (Figure 4.1) did surprisingly well in hiding the true value of the node degree (Figures 5.13 and 5.14). The true degree was only revealed in 5% of the attacks, and the property did not scale with the network domain size. At 50% mass of the experiments (Figure 5.14), the relative error was 50%, 80%, and 83% for the small, medium, and large topologies respectively. Better inherent protection would be expected if no high degree nodes were placed near the gateways of the Domain 2 topology.

From the privacy results the prototype digest algorithm provided significant protection against attacks to learn the sensitive properties evaluated. Using the attack heuristic to learn information from the digests yielded fairly uniform results irregardless of the domain topology specific composition and size. This low deviation in results is reflected in the low gSTD values in Table 5.2 and the plots in Figures 5.9 - 5.14. The rMSE growth of the estimates (Table 5.1) as the domain size increases further demonstrates that the attack reveals no more information about a large domain than it does about a small domain. The results further suggest that a privacy metric

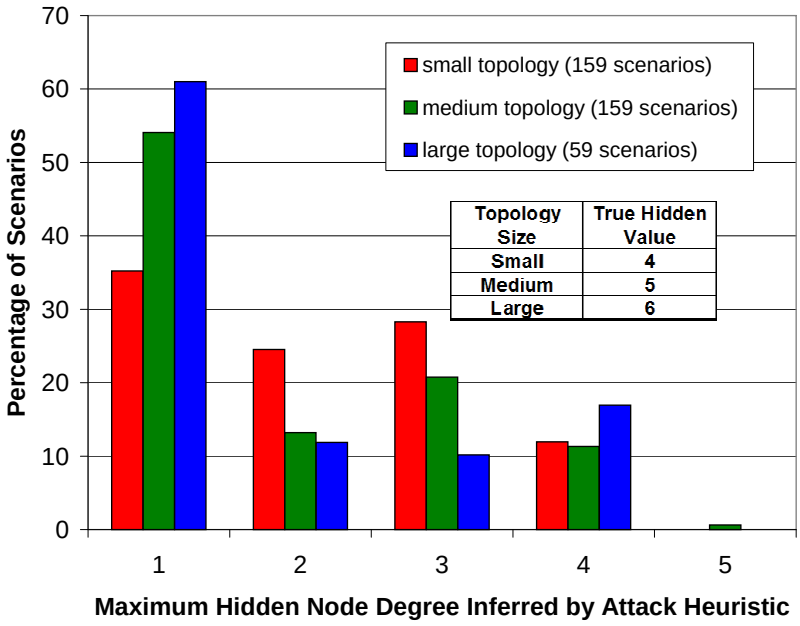


Figure 5.13. Histogram for the node degree property.

	Small Topology	Medium Topology	Large Topology
E	1.00	2.81	4.84

Table 5.3. SHRINK scalability results.

whose true value naturally grows with the sheer size of a domain receives inherent protection using a digest approach as the size of a network domain scales up. The network diameter and the number of routers naturally grow with a network domain's size, while a high degree node or an interior path between two gateways remains fairly static: an attack either finds it, or it does not.

4. Scalability Evaluation Results - SHRINK

To compute scalability E the average elapsed real time to compute SHRINK results for up to three failures was measured for the small, medium, and large topologies.

As expected, the SHRINK running time increased significantly as the number

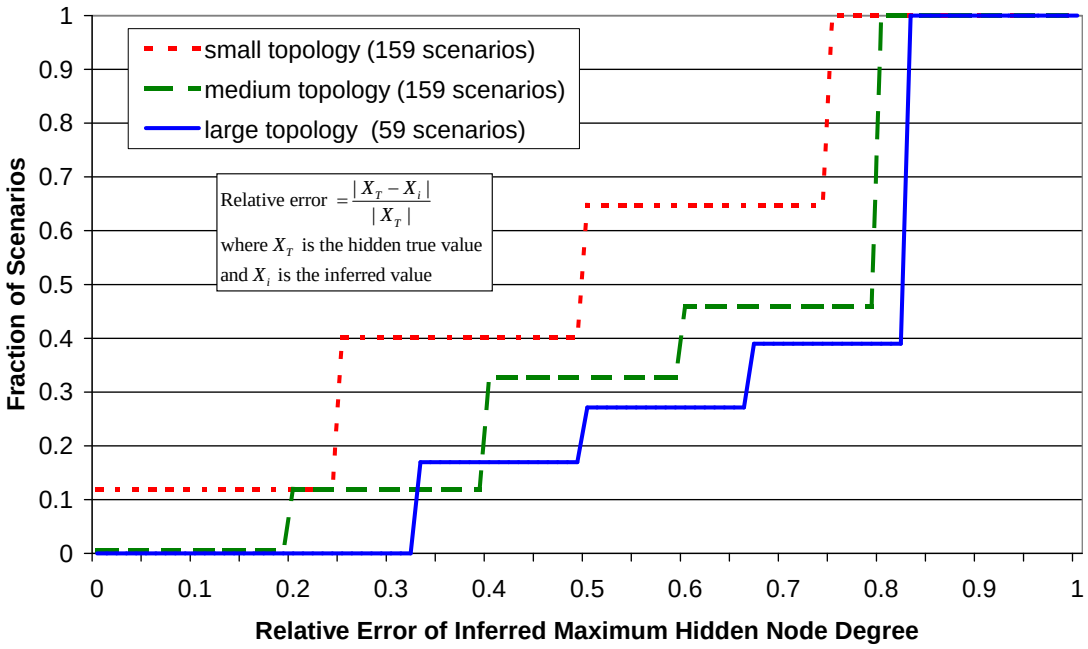


Figure 5.14. CDF relative error for the node degree property.

	Small Topology	Medium Topology	Large Topology
E	0.14	0.34	1.09

Table 5.4. SCORE scalability results.

of SRGs increased. The increase in scalability E by using the graph digest approach is evident in Table 5.3. Of particular note, inference time improved from hours to milliseconds on the large topology.

5. Scalability Evaluation Results - SCORE

To compute scalability E the average elapsed real time to compute the SCORE results for five threshold settings was measured for the small, medium, and large topologies respectively.

Although the SCORE running time increased less dramatically than SHRINK as the number of SRGs increased, a greater growth with full disclosure is evident than with the graph digest approach. The increase in scalability E is shown in Table 5.4.

B. PEER-PEER DOMAINS

1. Accuracy Evaluation Results - SHRINK

The initial inference results were puzzling as the α_s, α_u, and α_d scores were all low. SHRINK [19] tends to omit point-to-point links and stub routers from multiple failure scenarios using the default settings, instead attributing the evidence of failure about these components to an error in the SRG database. Although merely a nuisance in the provider-customer setting, the problem became magnified in the peer-peer domain setting due to the large number of links identified as cross-domain SRGs (e.g. the peering links and links on the web service shortest paths). Additionally, failures with low probability mass in one domain caused ambiguous inference results for B_i in the other domain. The SHRINK model implemented did not include a method for the inference to return $B_i = \varnothing$, which became a necessary feature in the peer-peer domain setting.

To counter the issue of SRG omission, the prior probabilities of the SRG nodes were lowered from 10^{-5} to 10^{-3}. After the change the inference engine preferred to add an additional SRG first, and assume an incorrect SRG database mapping second. To correct the null hypothesis problem, a low probability "Not I" node was implemented which indicates no failures internal to a domain. Using the low probability node is consistent with SHRINK.

The accuracy improvement metric A for the peer-peer topologies is depicted in Figures 5.15 and 5.16. In all but 1 of 484 tested cases, $\alpha_d \geq \alpha_s$, resulting in non-negative accuracy improvement scores A. In the tested scenarios, a minimum accuracy improvement of 31%, 30%, and 41% was observed for the small, medium, and large topologies respectively (highlighted with an oval in Figure 5.16). The A score average was 0.09, 0.062, and 0.124 and maximum value was 1.0, 0.33, and 0.5 for the small, medium, and large topology respectively. The results indicate that scaling the domain size has little impact on the accuracy of B_d, B_i, or A with respect to B_T. A slightly greater improvement in the large topology was seen, attributed to the rich

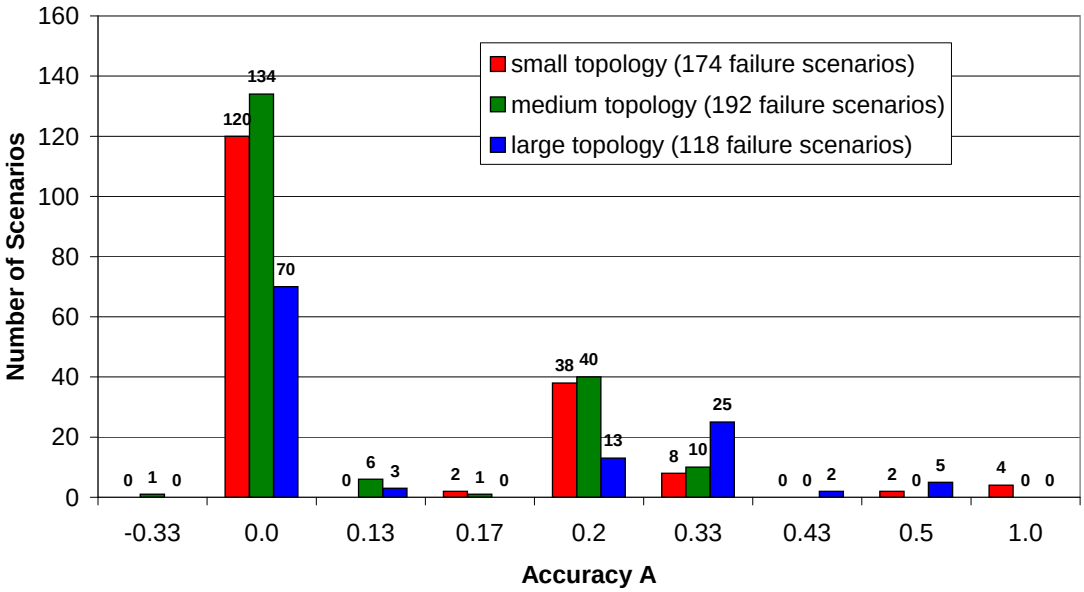


Figure 5.15. Histogram of A metric for the peer-peer setting.

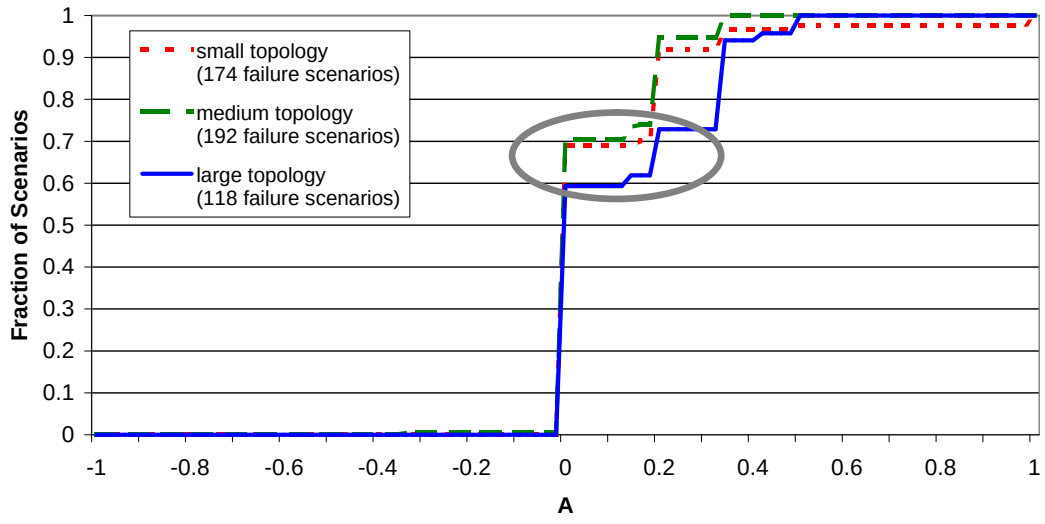


Figure 5.16. CDF of A metric for the peer-peer setting.

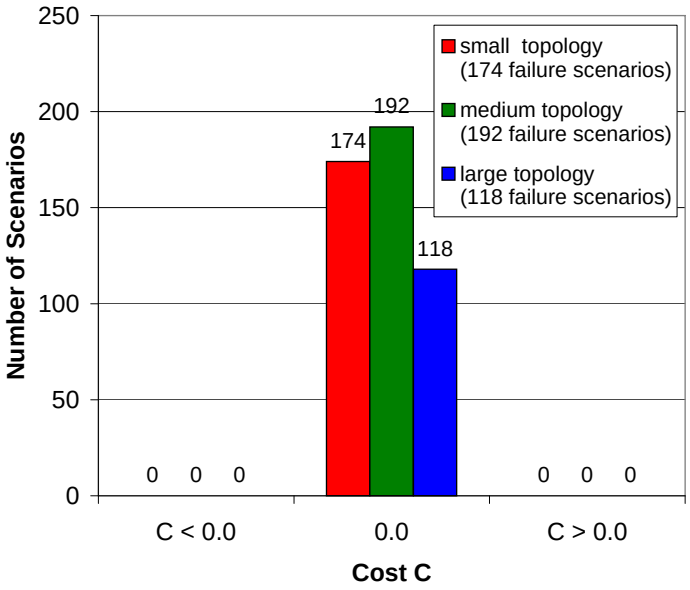


Figure 5.17. Histogram of C metric for the peer-peer setting.

number of cross-domain web service connections.

The negative accuracy A score occurred in the double-failure scenario $\{R_{37}, P_{26-37}\}$, where P_{26-37} represents the point-to-point link between routers $R26$ and $R37$. Domain 1 returned the cross-domain link $\{P_{26-37}\}$ as the best explanation. Domain 2 returned gateway router $\{R_{37}\}$ and a shared attribute for the cross-domain link in its causal graph. Using isolated inference, each domain correctly identified the failed component in its domain. The graph digest and full disclosure approaches both identified $\{R_{37}\}$ as the best explanation, preferring to treat evidence of failure about P_{26-37} as an error in the causal graph mapping.

As shown in Figures 5.17 and 5.18, the cost metric C was 0.0 (no cost) for all failure scenarios. The results mean that for all 484 tested failure scenarios, the digest approach achieved the same inference results as the full disclosure approach.

The Z values using the Wilcoxon Signed-Rank Test were 6.39, 6.19, and 6.01 for the small, medium, and large topologies respectively. These results each provide 95% confidence in the hypothesis.

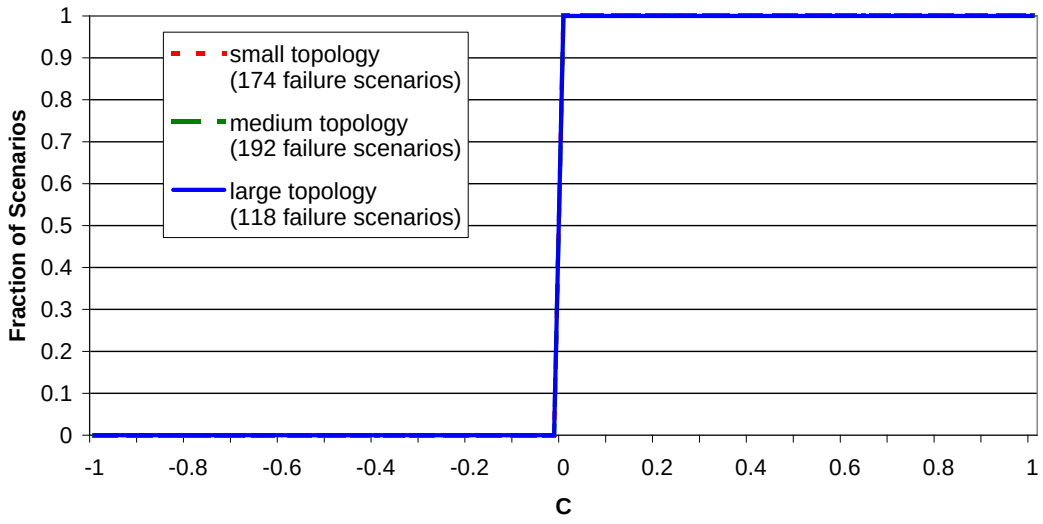


Figure 5.18. CDF of C metric for the peer-peer setting.

2. Accuracy Evaluation Results - SCORE

The accuracy improvement metric A for the peer-peer topologies is depicted in Figures 5.19 and 5.20. For all but one of the 484 tested cases, $\alpha_d \geq \alpha_s$, resulting in non-negative accuracy improvement scores A. The instance with a negative score

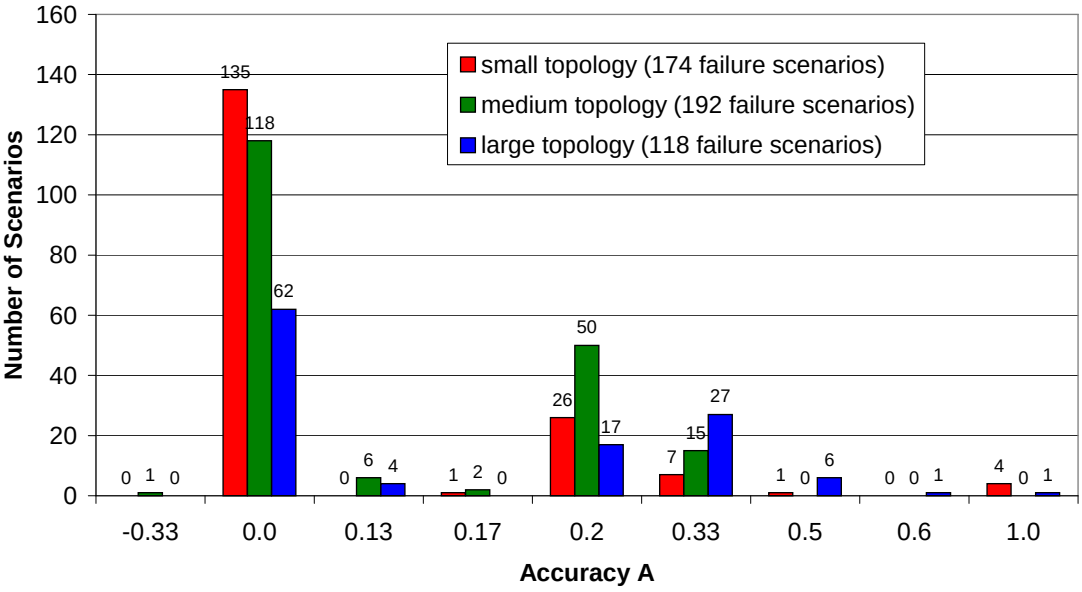


Figure 5.19. Histogram of A metric for the peer-peer setting.

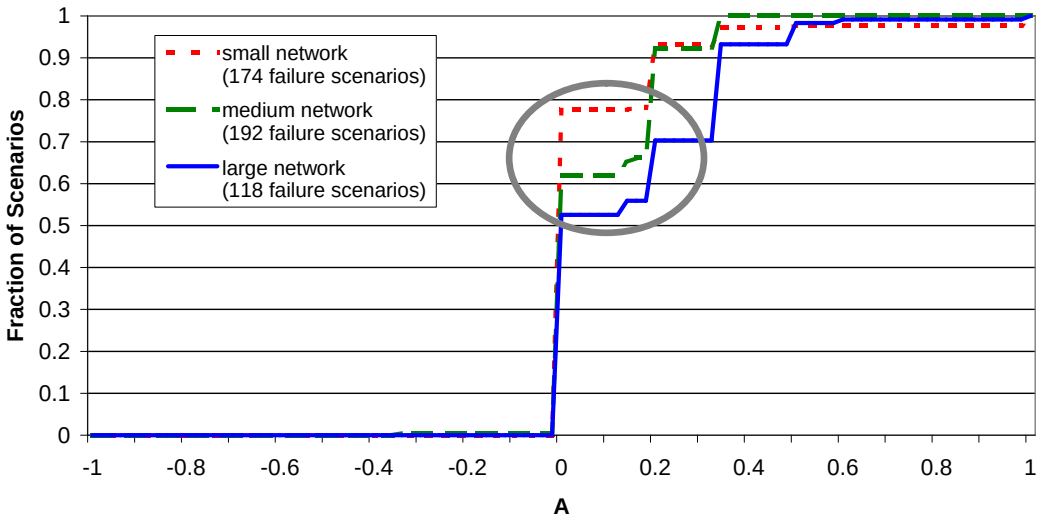


Figure 5.20. CDF of A metric for the peer-peer setting.

occurred in the same failure scenario, with the same details, as discussed above for SHRINK in Section 1. In the tested scenarios, an accuracy improvement of 22%, 38%, and 47% was observed in the small, medium, and large topology respectively. (highlighted with an oval in Figure 5.20). The A score average was 0.07, 0.08, and 0.15 and the maximum value was 1.0, 0.33, and 1.0 for the small, medium, and large topology respectively. The results indicate a trend that accuracy A increases as the domain size scales up. This result is attributed to the rich number of cross-domain web service connections.

As shown in Figures 5.21 and 5.22, the cost metric C was 0.0 (no cost) for all failure scenarios. The results mean that for all 484 tested failure scenarios, the digest approach achieved the same inference results as the full disclosure approach.

The Z values using the Wilcoxon Signed-Rank Test were 5.44, 7.12, and 6.51 for the small, medium, and large topologies respectively. These results each provide 95% confidence in in the hypothesis.

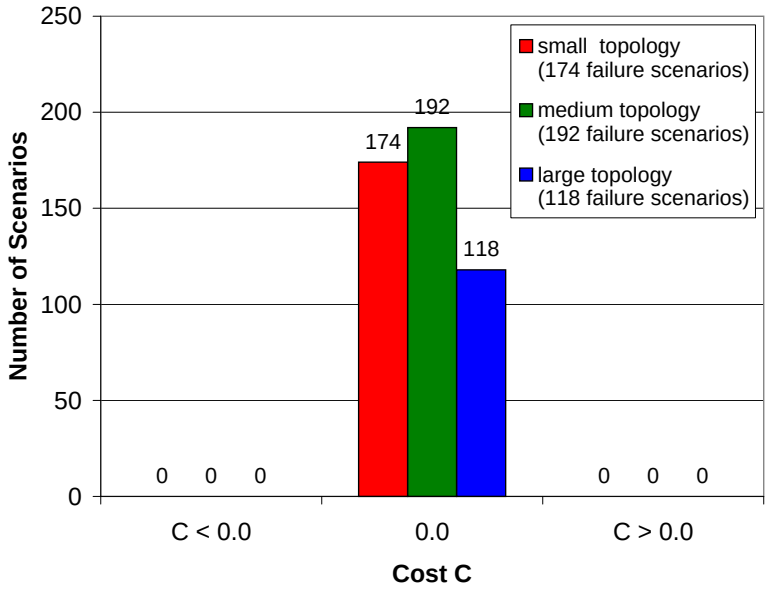


Figure 5.21. Histogram of C metric for the peer-peer setting.

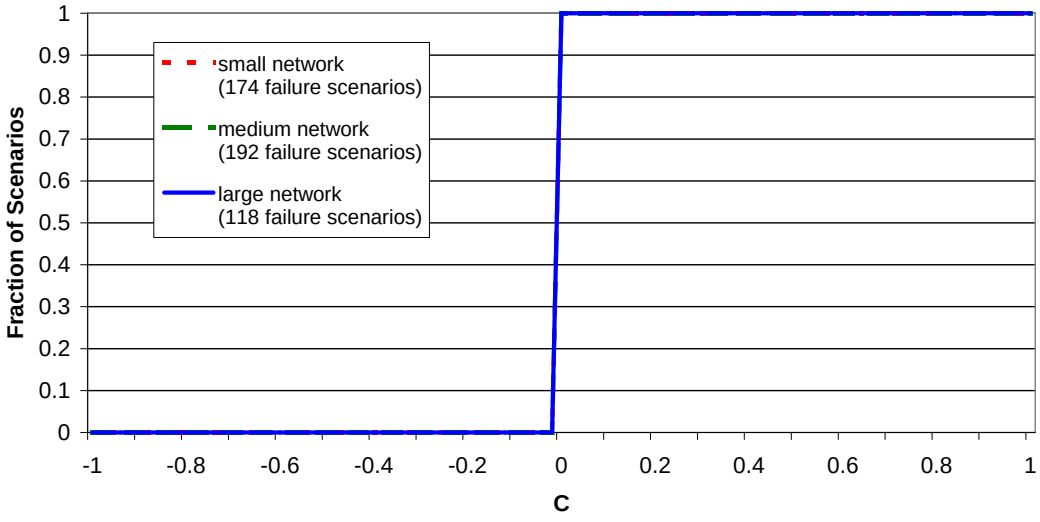


Figure 5.22. CDF of C metric for the peer-peer setting.

	Small	Medium	Large
Degree	2.63 (4)	4.64 (6)	5.79 (7)
Diameter	3.04 (5)	8.16 (10)	22.48 (24)
Routers	6.61 (10)	22.94 (26)	119.24 (122)

Table 5.5. Privacy metric rMSE versus (true value).

	Node Degree	Domain Diameter	Number Routers
gSTD	1.04	0.62	1.66
E(X)	1.46	1.83	3.19

Table 5.6. Privacy metric gSTD versus sample mean.

3. Privacy Evaluation Results

The privacy results using SHRINK and SCORE were identical for the reasons discussed in Section 3. The results below apply to digests created for both SHRINK and SCORE.

The root mean squared error results are shown in Table 5.5. Since the rMSE values are high relative to the true value, the information about the sensitive properties learned from the attacks results are generally far from the true values. The results from both the provider-customer and peer-peer settings are encouraging, and a more robust digest creation algorithm can surely improve on the results achieved by the prototype algorithm.

As depicted in Table 5.6, the generalized standard deviation for each privacy metric was low compared to the mean. As in the provider-customer setting, there was little variation in the amount of information learned about each sensitive property from each digest attack. An attacker using the attack heuristic will generally estimate similar sensitive property measurements across a range of failure scenarios and topologies.

Next, additional privacy protection data is provided by presenting histograms of the raw estimates and the relative error of the attack results against the true values

for each sensitive property.

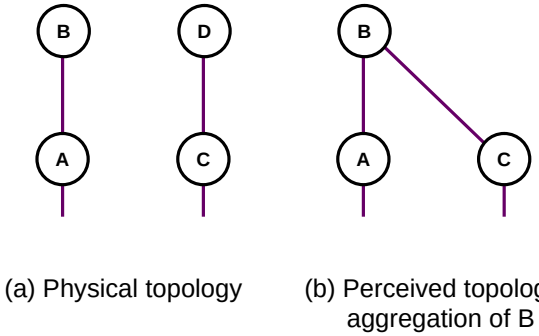


Figure 5.23. Aggregation affect on reachability.

Internal reachability between the visible gateways was revealed in 4 of the 484 evaluated peer-peer failure scenarios. The tested gateways are three, three, and two hops distant in small, medium, and large topologies respectively. Each instance of revealing the reachability occurred in the medium topology between gateways $R9$ and $R12$, which are 3 hops apart. The failed components were not in the neighborhood of the evaluated gateway routers as was the case with in provider-customer setting.

In each case of revealed reachability, aggregation by the digest algorithm (Figure 4.1) collapsed nodes and edges together that could individually reach one of the gateways. These aggregated nodes effectively created a bridge to establish reachability. To illustrate, gateway routers A and C connect to internal routers B and D respectively as shown in Figure 5.23(a). Gateways A and C may or may not actually be able to reach each other internally. If nodes B and D, representing the internal routers, are indistinguishable to the digest creation algorithm (Figure 4.1), they are aggregated into a single root cause node. The resulting topology after conducting a reverse-engineering attack on the digest using the heuristic in Figure 4.23 returns the topology shown in Figure 5.23(b). Digests containing revealed reachability would not pass the local security check shown in Figure 3.1 and would therefore not be distributed.

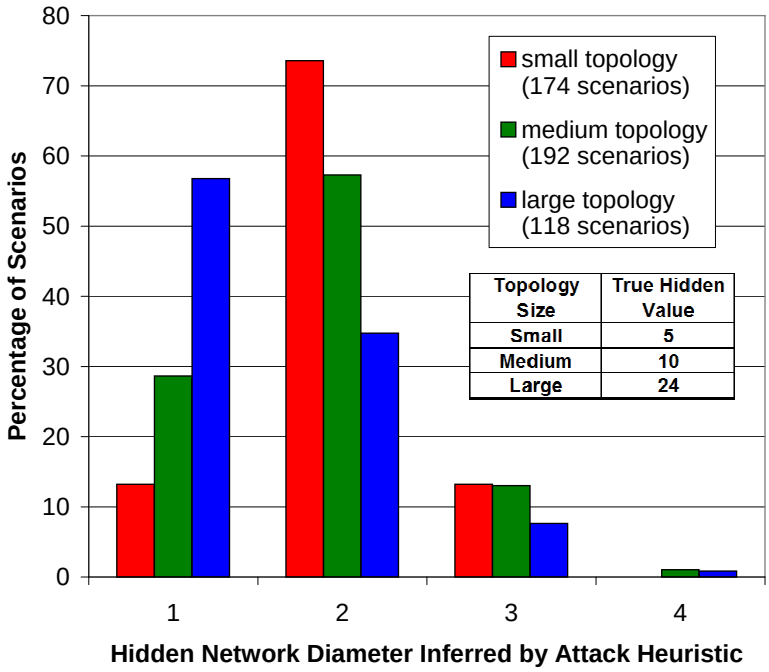


Figure 5.24. Histogram for the diameter property.

As in the provider-customer setting, the attack heuristic infers a narrow range of diameter estimates, irregardless of the actual domain network diameter as shown in Figure 5.24. The relative error between the true network diameter and the attack estimate grew with the topology size as shown in Figure 5.25. At 50% mass of the experiments the relative difference was approximately 60%, 80%, and 96% for the small, medium, and large topologies respectively.

The size of the topology had little bearing on the estimated number of routers as seen in Figures 5.26 and 5.27.

The three results with the highest inferred values were 9 routers in the medium topology and two results that found 10 routers in the large topology. One of the cases in which 10 routers were revealed in the large topology occurred when a high degree router failed in Domain 2. the digest creation algorithm (Figure 4.1) generated a failed IP link observation node for each link connected to the failed router. This failure scenario ($\{R98, P_{1-2}\}$) also revealed the true value of the high degree node in

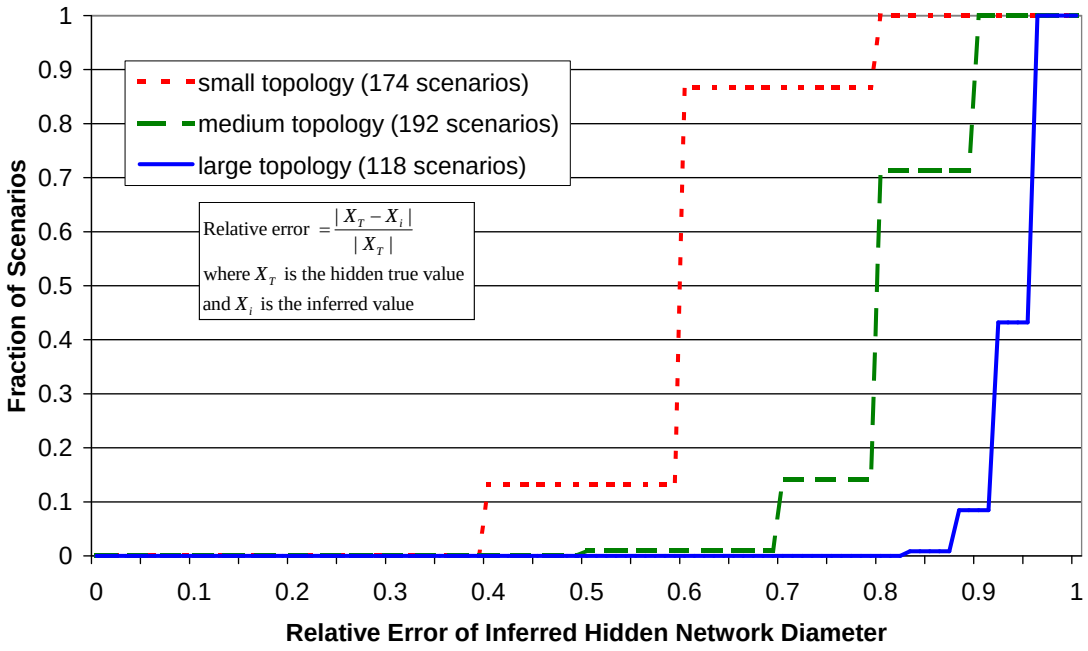


Figure 5.25. CDF relative error for the diameter property.

Figure 5.28. The other two results with the highest inferred values occurred in double failure scenarios when a link on the shortest path of many web service connections failed. The difference between the true value and attack estimate for 50% mass of the experiments (Figure 5.27) was 70%, 88%, and 98% for the small, medium, and large topologies respectively.

Privacy protection for maximum node degree scaled slightly in the peer-peer domain relationship (Figures 5.28 and 5.29) due to several nodes of higher degree in the internal topology of domain D_2. The node aggregation and Noisy-OR steps of the digest creation algorithm contributed to hiding the true value of the highest-degree node for most of the attacks, and only 1% of the attacks revealed the true high node degree. The difference between the true value and attack estimate for 50% mass of the experiments (Figure 5.29) was 75%, 83%, and 86% for the small, medium, and large topologies respectively.

Again inherent protection for the evaluated privacy metrics is seen that return similar results irregardless of the size of the domain topology. A stronger digest

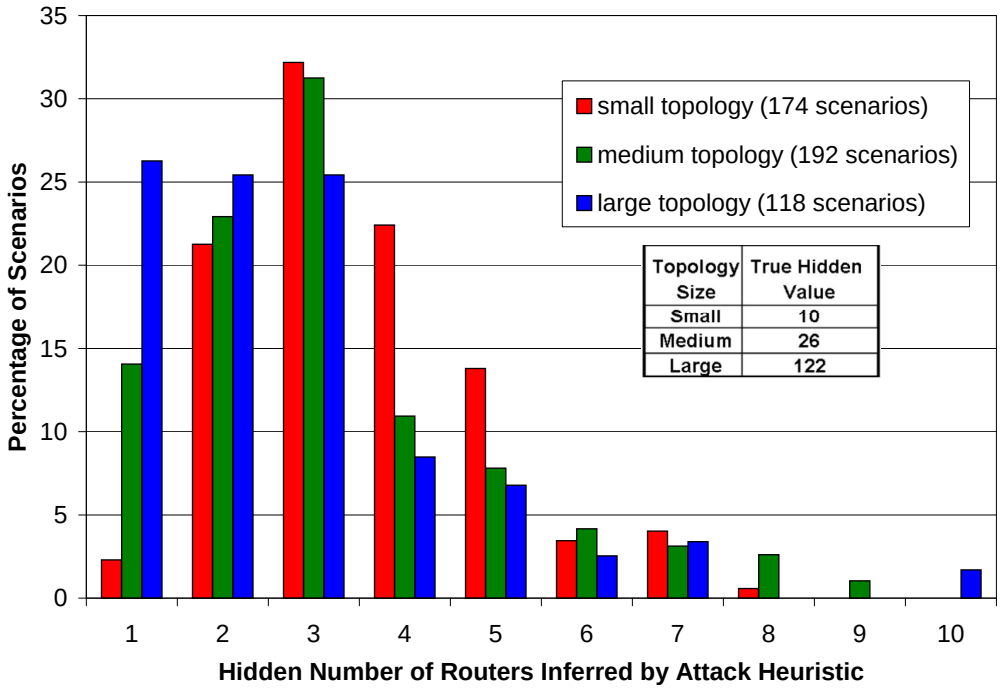


Figure 5.26. Histogram for the number of routers property.

	Small Topology	Medium Topology	Large Topology
E	0.71	1.29	1.72

Table 5.7. SHRINK scalability results.

algorithm and post-processing of a digest to remove any information over a predesignated threshold will intuitively strengthen a digest against entropy loss to attack. Several digests were created that failed to sufficiently hide network sensitive properties, and these digests would not be distributed. The node aggregation step of the digest algorithm (Figure 4.1) strips information from a graph digest, but with some unintended consequences as discussed above. A more robust version of the algorithm should address the identified shortcomings.

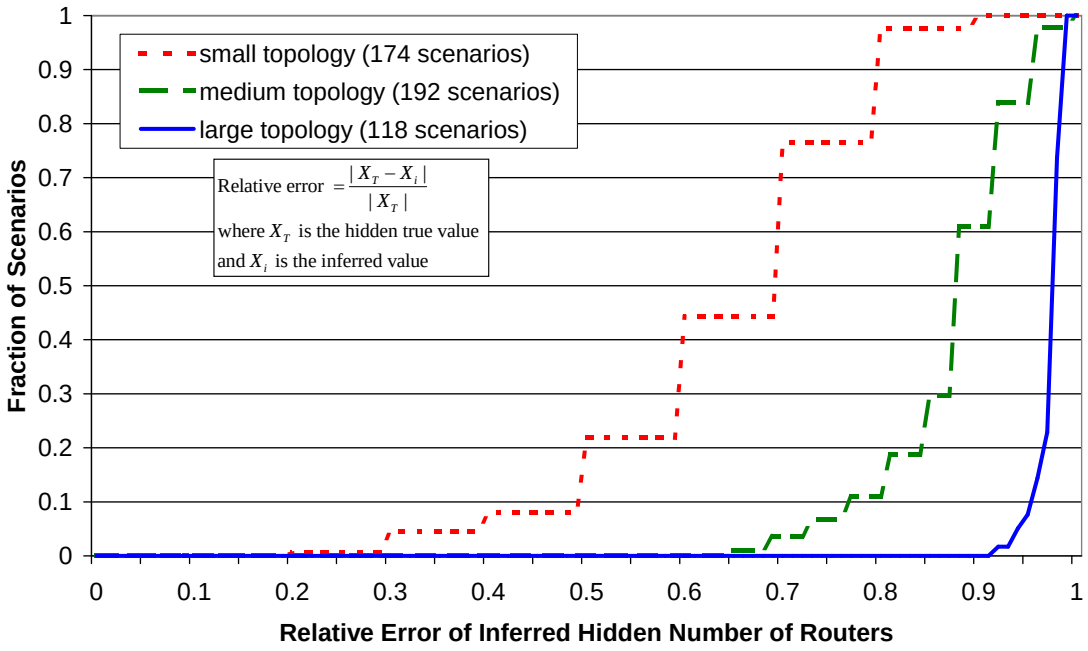


Figure 5.27. CDF relative error for the number of routers property.

	Small Topology	Medium Topology	Large Topology
E	0.07	0.18	0.62

Table 5.8. SCORE scalability results.

4. Scalability Evaluation Results - SHRINK

The scalability (speed) improvement for the peer-peer domain scenario (Table 5.7), while significant, is not as dramatic as that observed in the provider-customer setting. In the peer-peer scenario the domain performing inference has a larger structure, resulting in a greater number of hypotheses for the inference engine to consider. While not as pronounced as in the provider-customer setting, the running time savings are still significant.

5. Scalability Evaluation Results - SCORE

As in the provider-customer setting, scalability E (Table 5.8) improvement is not as dramatic as that realized with SHRINK. Clearly though, using the graph

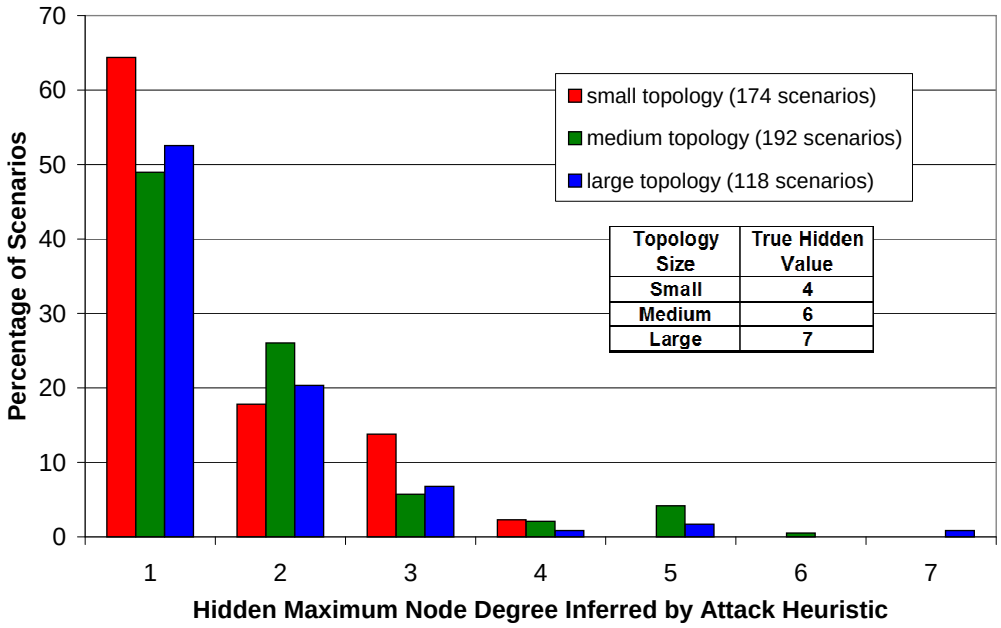


Figure 5.28. Histogram for the node degree property.

digest approach achieves much faster inference running time than the full disclosure approach.

C. ABILENE-BASED SETTING

1. Accuracy Evaluation Results

For all but 5 of 377 tested scenarios, $\alpha_d \geq \alpha_s$, resulting in non-negative accuracy improvement scores A (Figure 5.30). The average score was 0.25 for both SHRINK and SCORE, and the maximum score for each was 1.0. There was an accuracy improvement in 34% and 68% of the test scenarios for SHRINK and SCORE respectively (indicated by an oval in Figure 5.31).

All instances of accuracy $A = 1.0$ in Figure 5.30 reflect failure scenarios for which at least one component in the provider domain (Domain 1) failed. In these failure scenarios all failures were in the provider domain, or one failure occurred in the provider domain and either a point-to-point link or a stub router failed in the customer domain. In the latter cases, with the default SHRINK settings a best

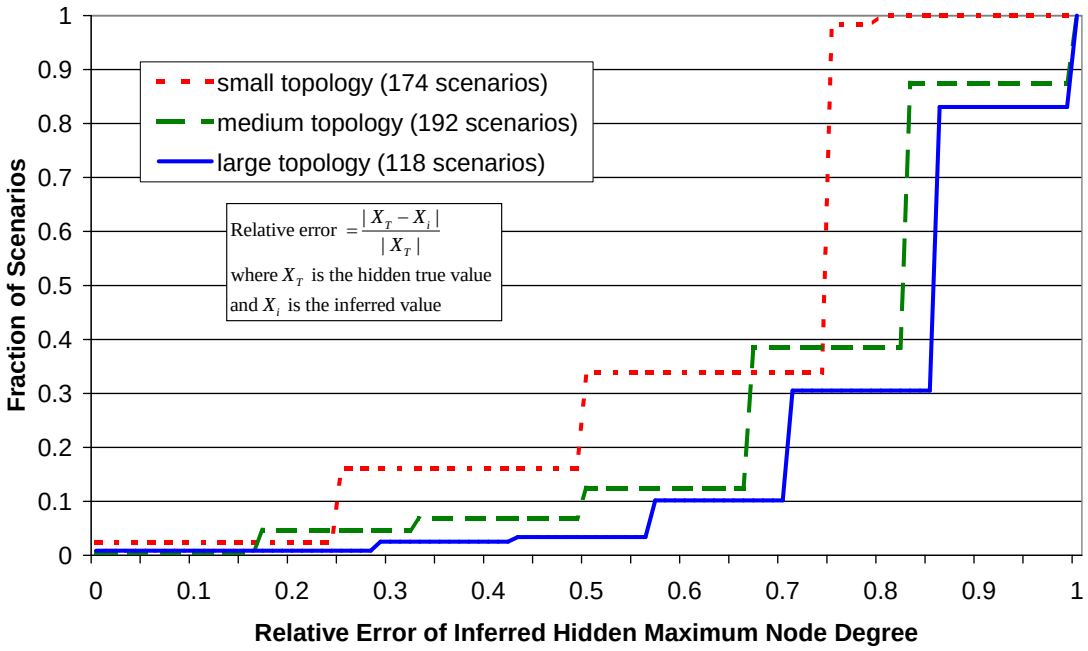


Figure 5.29. CDF relative error for the node degree property.

explanation based on a single incorrect SRG dependency mapping may have a higher posterior probability than a best explanation containing one more failed SRG.

The five negative accuracy results occurred using SCORE. The results stem from double-failure scenarios containing a gateway in the customer domain and a non-adjacent component in the provider domain. Each of these failure scenarios resulted in all three leased circuits failing, and identifying the provider as the root cause of all observed failures. These failure scenarios highlight a shortcoming in the greedy heuristic used by SCORE. SHRINK, which returns the hypothesis with the maximum posterior probability, correctly identified the failed gateways in these failure scenarios.

The cost metric C depicted in Figure 5.33, shows no cost in using the graph digest approach. The cost to inference accuracy equaled zero in all test cases, meaning that the digest approach achieved the same accuracy as the full disclosure approach in all tested scenarios.

The Z values using the Wilcoxon Signed-Rank Test were 6.39 and 9.18 for SHRINK and SCORE respectively. The hypothesis passes the 95% confidence test

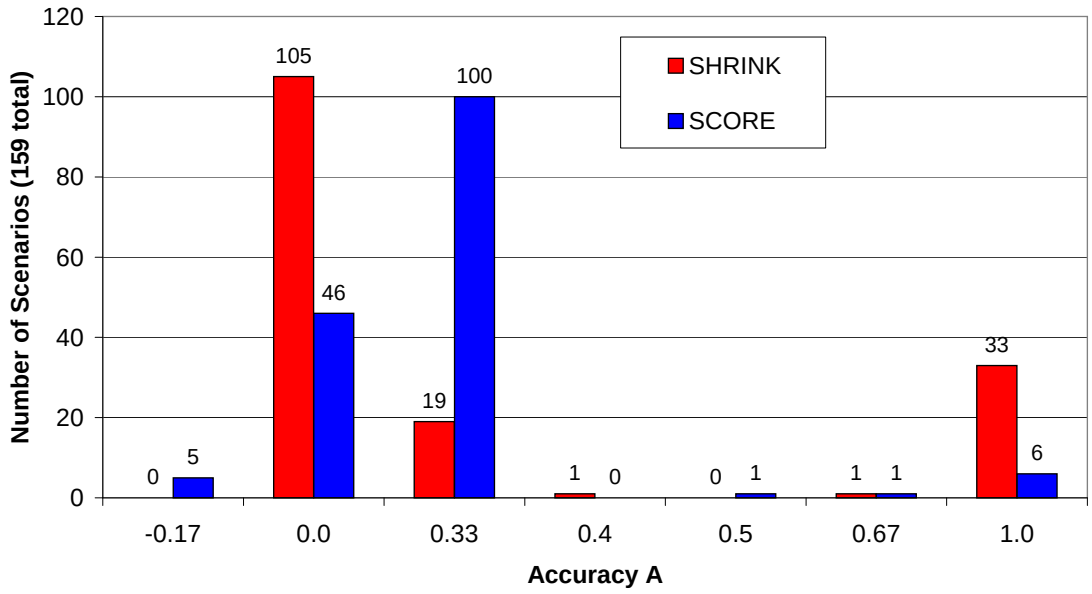


Figure 5.30. Histogram of A metric for the Abilene-based topology.

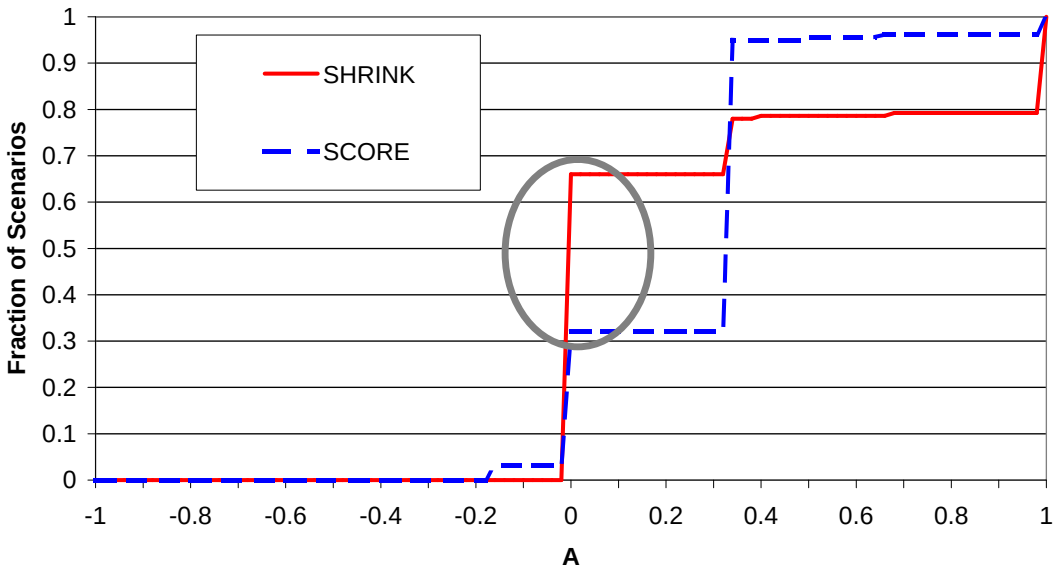


Figure 5.31. CDF of A metric for the Abilene-based topology.

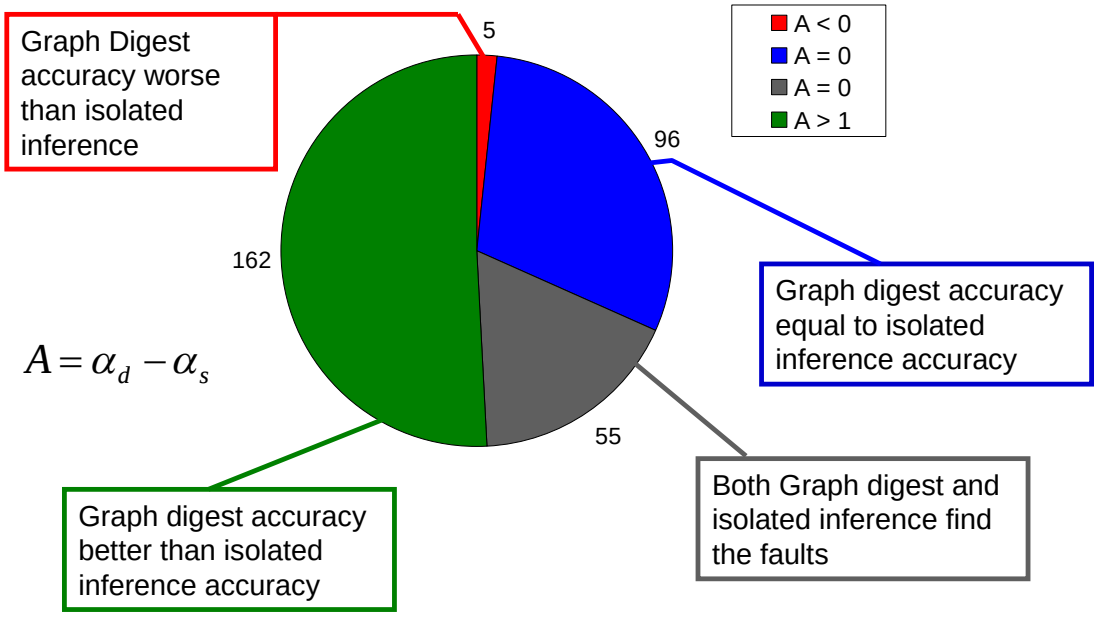


Figure 5.32. Summary of A metric results for the Abilene-based topology.

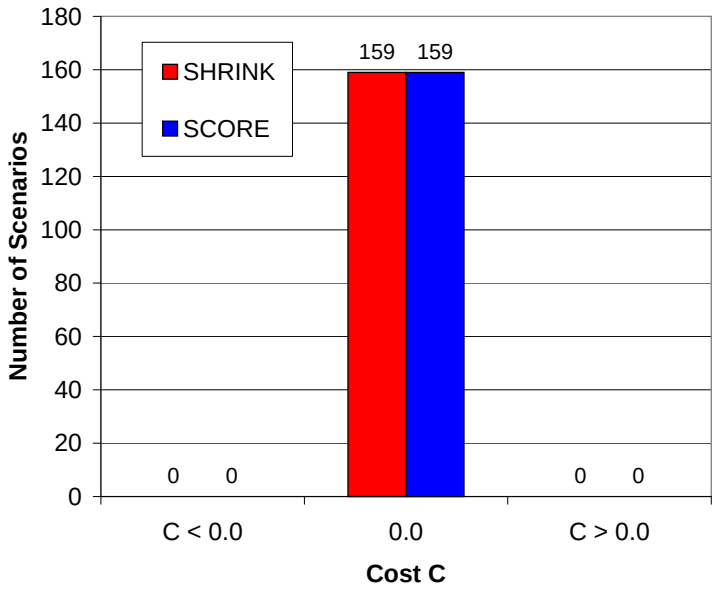


Figure 5.33. Histogram of C metric for the Abilene-based topology.

Degree	4.02 (6)
Diameter	6.56 (8)
Routers	67.36 (71)

Table 5.9. Privacy metric rMSE versus (true value).

	Node Degree	Domain Diameter	Number Routers
gSTD	1.41	1.00	2.23
E(X)	2.24	1.52	3.68

Table 5.10. Privacy metric gSTD versus sample mean.

for both inference algorithms.

2. Privacy Evaluation Results

The rMSE values of the attack estimates were close to the true hidden values as depicted in Table 5.9. The gSTD values were low compared to the mean attack values as shown in Table 5.10. These results indicate that the digest attacks were unsuccessful at revealing the hidden values for the sensitive properties.

Histograms and CDFs of the attack estimates for each sensitive property, less reachability, are presented in Figures 5.34 - 5.39. The reachability test was conducted between gateway routers $R4$ and $R7$ in Figure 4.17. The gateways are 3 hops apart via internal links, and none of the 159 digests revealed the hidden reachability between these two gateway routers.

The network diameter values measured by the attack are presented in Figure 5.34. At 50% mass of the experiments (Figure 5.35), the relative error is approximately 88%.

A histogram showing the number of routers found by the attack heuristic is presented in Figure 5.36. At 50% mass of the experiments (Figure 5.37), the relative error is approximately 97%. Intuitively this results makes sense since each digest only provides a small collection of nodes. In general the topology learned consists of

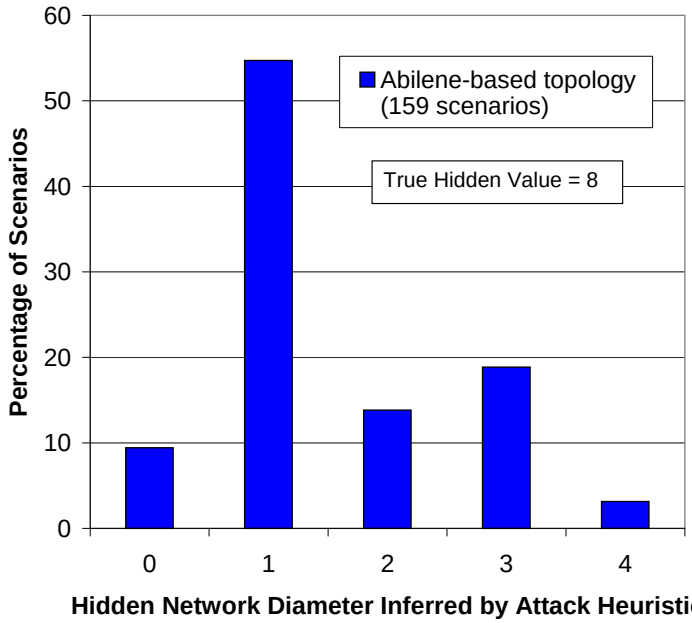


Figure 5.34. Histogram for the diameter property.

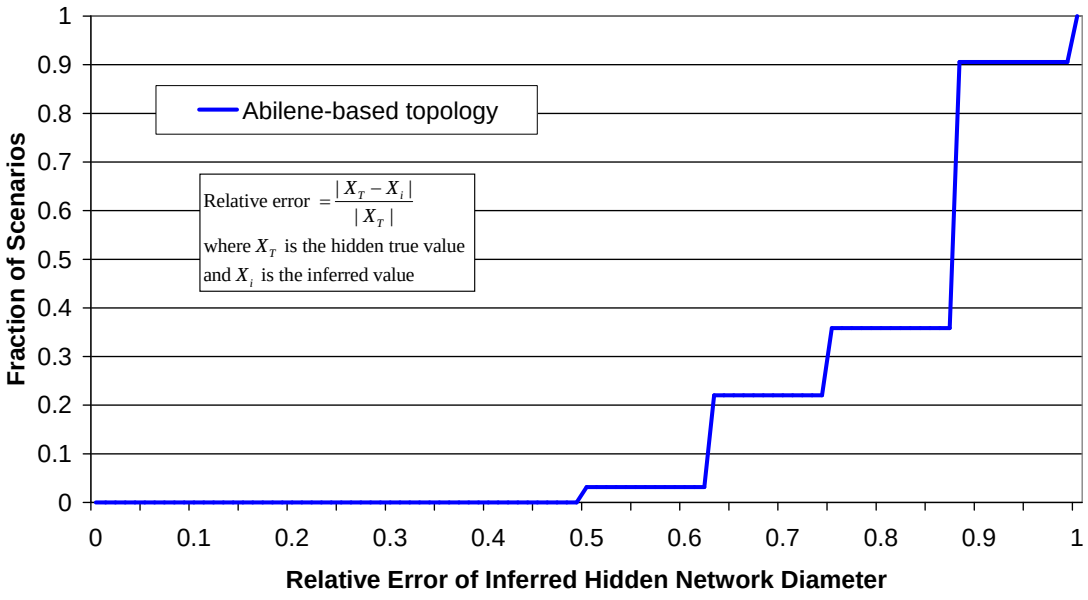


Figure 5.35. CDF relative error $|X_T|$ for the diameter property.

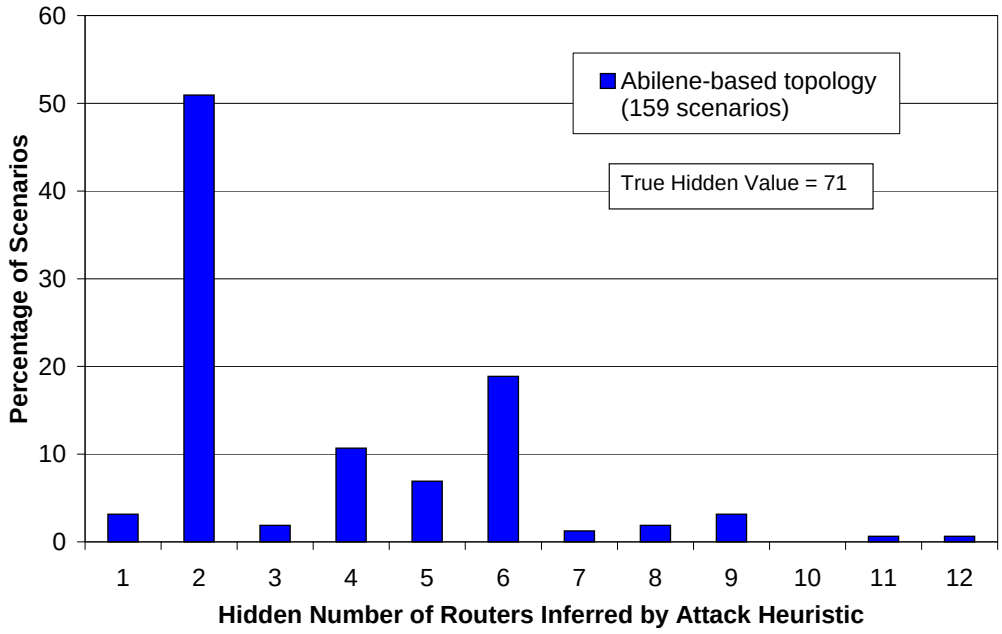


Figure 5.36. Histogram for the number of routers property.

	SHRINK	SCORE
E	2.47	0.24

Table 5.11. SHRINK and SCORE scalability results.

a neighborhood around one or two routers, and multiple failures whose neighborhoods intersect allow a larger portion of the topology to be inferred.

A histogram depicting the maximum node degree attack results is shown in Figure 5.38. At 50% mass of the experiments (Figure 5.39), the relative error was approximately 67%.

3. Scalability Evaluation Results

To compute scalability E the average elapsed real time to compute SHRINK and SCORE results for up to three failures was measured on the Abilene-based topology. The mean elapsed real time of the first ten double failure scenarios for both full disclosure inference and for the graph digest approach was used to compute E.

As expected the graph digest approach enabled significant time savings while

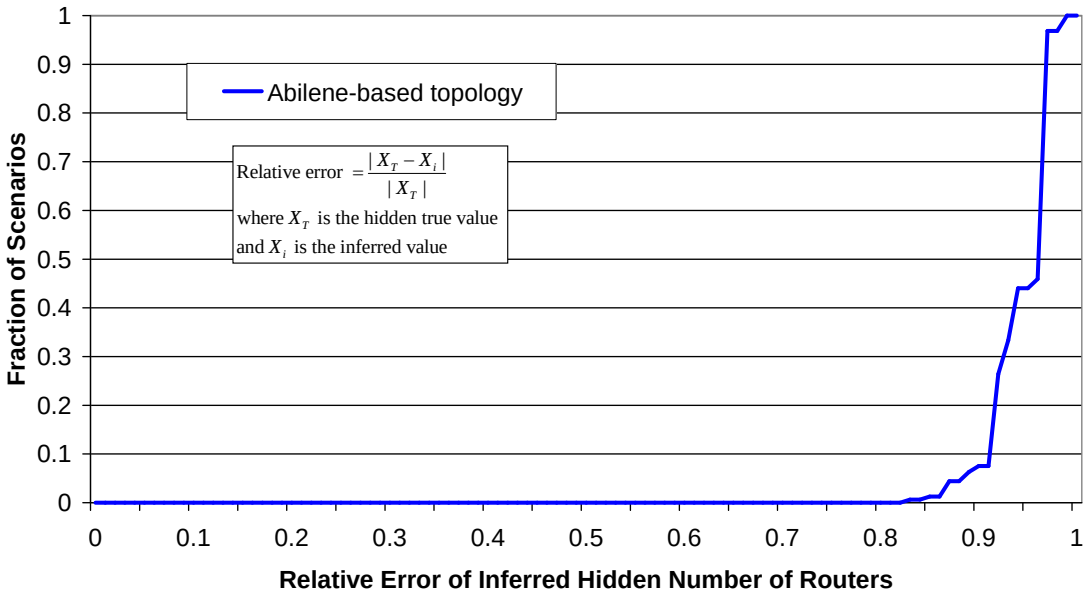


Figure 5.37. CDF relative error for the number of routers property.

performing inference with SHRINK. Although not as dramatic, the graph digest approach enabled faster inference using the SCORE algorithm as well. These results are consistent with the scalability results on the synthetic topologies constructed with network topology atoms.

D. CONCLUSION

This chapter provides the experimental results using the evaluation methodology outlined in Chapter IV. For each of the seven tested topologies and for both implemented intra-domain fault localization algorithms, the hypothesis of accuracy improvement was supported with a 95% confidence level. Although the hypothesis testing was conducted for 95% confidence, all fourteen hypothesis tests would have passed at 99.95% confidence.

A summary of the accuracy results for the synthetic topologies is presented in Figure 5.40. In total, there were 814 failure scenarios in which using the graph digest approach improved accuracy in finding cross-domain faults.

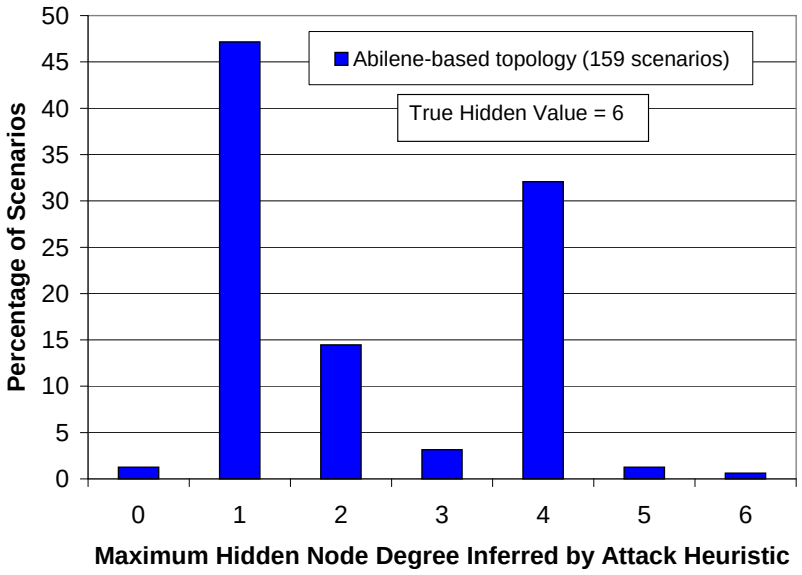


Figure 5.38. Histogram for the node degree property.

In general, both SHRINK and SCORE performed well at finding faults. When a highly connected device failed, all approaches typically correctly identified the failed component using SHRINK or SCORE. Using the full disclosure approach, SCORE with a mean accuracy of 0.97 outperformed SHRINK with a mean accuracy of 0.88 . Using the graph digest approach, SHRINK with a mean A score of 0.21 outperformed SCORE with a mean A score of 0.19. Using the default SHRINK settings, SHRINK assigns a lower prior probability of failure to a device than the probability that a conditional dependency mapping is incorrect. SCORE tended to outperform SHRINK when a stub router or point-to-point link failed. SHRINK with its built-in robustness to errors in causal graphsperformed better at performing inference on transformed data that included a graph digest.

The privacy results show measurable protection of the sensitive properties evaluated. The estimate means were generally far from the true values with little deviation in the estimates. Clearly some digests, such as those revealing reachability, would fail local security policy checks. These digests would not be sent as discussed in Chapter III, Section B. The privacy results for the synthetic topologies and empirical

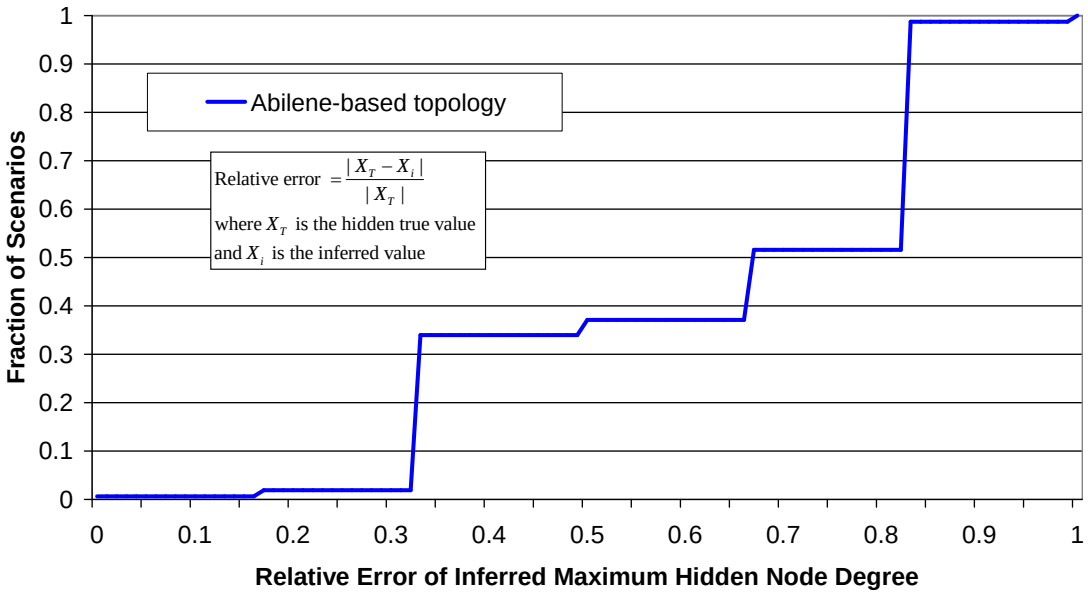


Figure 5.39. CDF relative error for the node degree property.

	Small	Medium	Large	Abilene-Based
Degree	2.11 (4)	3.28 (5)	4.33 (6)	4.02 (6)
Diameter	3.27 (4)	9.95 (11)	22.01 (23)	6.56 (8)
Routers	12.06 (15)	47.71 (51)	197.97 (201)	67.36(71)

Table 5.12. Summary of provider-customer rMSE versus (true value).

topology are similar, strengthening an argument for applicability of the graph digest approach to a broad range of networks. A summary of privacy rMSE and gSTD results for the provider-customer topologies is presented in Tables 5.12 and 5.13.

One of the surprises from the experiments is that the proposed metric $gSTD$ is much more effective than expected at gauging the performance of the digest-creation

	Node Degree	Domain Diameter	Number Routers
gSTD	1.20	0.74	1.85
E(X)	2.08	1.11	3.31

Table 5.13. Summary of provider-customer gSTD versus sample mean.

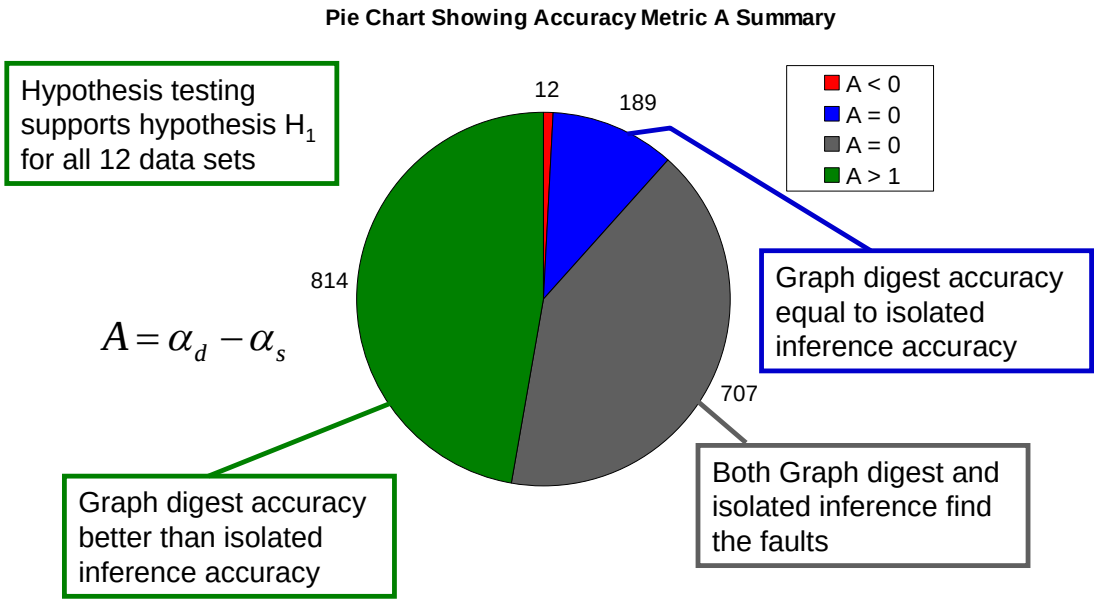


Figure 5.40. Summary of A metric for the synthetic topologies.

algorithm in hiding the values of sensitive properties. While further investigations are required to validate the generality of such effectiveness, the results give weight to further investigation into the approach.

A decrease in running time was demonstrated by using the graph digest approach versus full collaboration, achieving positive scalability results for the approach.

The experimental results reaffirm the observation that more research efforts in this space are needed. In all topologies simulated, a number of scenarios where domains cannot troubleshoot effectively in isolation were discovered . A large portion of the real world scenarios are expected to be more complicated than those evaluated in this research. Therefore, the need for cross-domain solutions is real.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSIONS

This chapter first provides the main conclusions from the research, and then discusses future work identified in the course of the research.

A. RESEARCH CONCLUSIONS

This research demonstrated that by correlating risks and observations from different domains, cross-domain fault localization has the potential to significantly increase the accuracy of network fault localization. It also articulated the main challenges to realize inference accuracy gain, particularly the privacy consideration. The main contributions are a framework with explicit metrics to evaluate a cooperative design in the design space, and an inference-graph-digest based formulation of the problem. The graph digest approach also facilitates the re-use of existing fault localization algorithms without compromising each domain's information hiding policy.

The evaluation supported the hypothesis with 95% confidence for all 14 evaluated data sets: *It is possible to construct a framework to enable managers of separate network domains to share information and achieve inference gain while quantifying privacy preservation of sensitive information.*

This research presents the first comprehensive evaluation of the feasibility of cross-domain fault localization. The evaluation is systematic and complete with regarding to all the proposed performance metrics.

The goal was to answer the following overarching questions:

1. Does cross-domain fault localization offer the kinds of benefits warranting further research?
2. Can it provide deployable and acceptable privacy protection with manageable complexity?

The answer to both of these questions was a strong "Yes". Cross-domain fault localization, using the prototype design, performed quite well at finding the faults

in all failure scenarios. Of course, in practice not using a design that balances the requirements of accuracy and privacy is a non-starter — domain administrators will be simply unwilling to reveal their complete topologies. This leads to the second question. The digest approach did provide significant performance gains compared with localization performed in isolation while measurably protecting the sensitive properties tested. The use of a design that enables sharing summary information dramatically increases the deployability of cross-domain fault localization by decreasing inference time by two to three orders of magnitude.

While providing a positive answer to both high-level questions, the evaluation also reveals several opportunities for further research and enhancement including richer causal graph models and better digest algorithms. This underscores the importance of having a repeatable evaluation methodology.

B. FUTURE WORK

To move forward certainly requires a fundamental understanding of the issues beyond the framework, approach, and scenarios described in this dissertation. Is the graph digest approach applicable to a wide range of network scenarios? What about scenarios involving more than two domains? Does there exist a general, yet easily calculable metric for quantifying the highly domain-specific information hiding policy? How should observation errors and graph model inaccuracies be detected and controlled? These and other similar questions constitute a new area of networking research which may have a major impact on network fault trouble-shooting practices.

In light of this work, a prioritized list of future work follows.

- Beyond visual checks of "Does this seem reasonable?", this research did not validate how well the scenarios capture issues faced in real world practice. There is little or no publicly available data to allow this validation. Further testing on real network topologies must be done as a logical next step to validate the utility of the framework and graph digest approach. Obtaining troubleshooting records and topology from one's own domain is challenging enough. Collecting

such sensitive data from multiple domains is almost impossible. There may be hope as data collection with fault logging by collaborative projects such as Abilene and GEANT would provide a much needed evaluation dataset for cross-domain fault localization, and fault localization in general.

- Distributions accurately representing an adversary's beliefs about sensitive properties that model general and specific events would allow application of the KL distance to evaluate privacy protection. While this may remain a lofty goal for many sensitive properties, any such distributions that can be established will enable use of the ideal metric for privacy protection.

- Provable practical privacy risk metrics are needed. The presented rMSE and gSTD practical privacy metrics provide a sound methodology, but their strengths and limitations still need to be proven. Domain managers will be reluctant to use any cooperative design without proven bounds on the risk to sensitive properties.

- An ontology of sensitive properties with privacy protection implementation methods is needed. While sensitive properties may vary between domains, there may a core set that most domain managers would agree comprise the majority of these properties. Identifying the common sensitive properties is a logical first step needed in order to develop robust protection against disclosure.

- Some inherent bias is acknowledged in attacking the digests using the attack heuristics developed in this research. The entire network structure of the undigested causal graphs is revealed using the attack heuristics, however, indicating a sound baseline attack method. While a more thorough attack strategy is needed, it may be possible to determine the effectiveness and limitations of the presented heuristic. A variant of the heuristic may be a valuable component of such a future attack strategy. This is just one of many reasons why rigorous analysis and proofs of privacy are needed before the graph digest approach can be widely adopted when privacy is a concern.

- The current algorithm for constructing digests incorporates network domain knowledge. Techniques from the artificial intelligence and statistics communities for approximating statistical distributions could be leveraged to produce smaller and more accurate digests. Since performing fault localization with digests is significantly faster than without, perhaps digests can be used internally in very large domains to yield faster inference.

  For example, instead of allowing a digestion algorithm to produce variable sized digest causal graphs, the size and/or structure of the graph digest *a prior* may be constrained, similar to the way in which a secure hash function has a predefined fixed width in bits (e.g., 512 bits for SHA-512) for all hash

values. The primary advantage of this approach is that the $gSTD$ would be small regardless of the scenarios used. However, this approach also brings up a challenge. By restricting the size and structure of a graph digest, it might be difficult to encode within it sufficient information to support inference for large scenarios.

- Observation errors or missing observations when evaluating inference accuracies were not modeled in this research. Such events are common in the real-world due to software bugs or misconfigurations. Follow-on work should evaluate the performance of the graph digest approach in a noisy observation environment. These errors are expected to similarly impact all discussed approaches and, therefore, introduce very small perturbations to the A and C metrics.

- The core network causal graph model (SHRINK [19]) has a very simple structure. The structure has the advantage of easy inference but lacks expressiveness. In particular, the bipartite nature makes compositing levels difficult. The Sherlock [4] work gives a more expressive model without sacrificing SHRINK's inference speed advantages. Expanding the expressive power of the causal graph model requires new algorithms for specifying shared attributes, combining graphs and for constructing digests.

- Finally, in addition to the development of better metrics and algorithms, an emphasis should be placed on the creation of *new theories* for reasoning about what can and cannot be achieved in balancing the trade-off between inference accuracy and privacy protection. Appropriate mechanisms, trust models, and policy must also be developed to support the exchange of causal graph digests and other relevant information (e.g., shared attributes) between domains in collaboration.

LIST OF REFERENCES

[1] Internet2 network, June 2009. `http://www.internet2.edu/network/`.

[2] M. K. Aguilera, J. C. Mogul, J. L. Wiener, P. Reynolds, and A. Muthitacharoen. Performance debugging for distributed systems of black boxes. *Proceedings of the nineteenth ACM symposium on Operating systems principles*, pages 74–89, 2003.

[3] Ricardo. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval*. Addison-Wesley Longman, 1999.

[4] P. Bahl, R. Chandra, A. Greenberg, D. A. Maltz, and M. Zhang. Towards highly reliable enterprise network services via inference of multi-level dependencies. *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 13–24, 2007.

[5] A. T. Bouloutas, S. B. Calo, A. Finkel, and I. Katzela. Distributed fault identification in telecommunication networks. *Journal of Network and Systems Management*, 3(3):295–312, 1995.

[6] Justin Brickell and Vitaly Shmatikov. Privacy-preserving graph algorithms in the semi-honest model. In *In ASIACRYPT, LNCS*, pages 236–252. Springer-Verlag, 2005.

[7] Rui Castro, Mark Coates, Gang Liang, Robert Nowak, and Bin Yu. Network tomography: recent developments. *Statistical Science*, 19:499–517, 2004.

[8] M. Y. Chen, A. Accardi, E. Kiciman, J. Lloyd, D. Patterson, A. Fox, and E. Brewer. Path-based faliure and evolution management. *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation-Volume 1 table of contents*, pages 23–23, 2004.

[9] Mark Coates, Alfred Hero, Robert Nowak, and Bin Yu. Internet tomography. *IEEE Signal Processing Magazine*, 19:47–65, 2002.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms, Second Edition*. The MIT Press Cambridge, Massachusetts, 2001.

[11] T. M. Cover, J. A. Thomas, J. Wiley, and W. InterScience. *Elements of Information Theory*. Wiley-Interscience New York, 2006.

[12] J. L. Devore. *Probability and Statistics for Engineering and the Sciences*. Brooks/Cole - Thomson Learning, California, 2004.

[13] D. D. Dimitrijevic, B. Maglaris, and R. R. Boorstyn. Routing in multidomain networks. *IEEE/ACM Transactions on Networking (TON)*, 2(3):252–262, 1994.

[14] N. Feamster, J. Borkenhagen, and J. Rexford. Guidelines for interdomain traffic engineering. *ACM SIGCOMM Computer Communication Review*, 33(5):19–30, 2003.

[15] W. Fischer, G. Xie, and J. Young. Cross-domain fault localization: A case for a graph digest approach. *Proceedings of IEEE Internet Network Management Workshop*, October 2008.

[16] William D. Fischer, Geoffrey G. Xie, and Joel D. Young. Is cross-domain fault localization feasible? Technical Report NPS-CS-09-007, Naval Postgraduate School, February 2009.

[17] Michael Hay, Gerome Miklau, David Jensen, and Siddharth Srivastava. Anonymizing social networks. Technical report, Science, 2007.

[18] X. Huang, S. Zou, W. Wang, and S. Cheng. Mdfm: Multi-domain fault management for internet services. *Management of Multimedia Networks and Services: 8th International Conference on Management of Multimedia Networks and Services, MMNS 2005, Barcelona, Spain, October 24-26, 2005: Proceedings*, 2005.

[19] S. Kandula, D. Katabi, and J. P. Vasseur. Shrink: a tool for failure diagnosis in ip networks. *Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 173–178, 2005.

[20] Hillol Kargupta and Souptik Datta. On the privacy preserving properties of random data perturbation techniques. In *In ICDM*, pages 99–106, 2003.

[21] I. Katzela, AT Bouloutas, and SB Calo. Centralized vs. distributed fault localization. *Proceedings of the fourth international symposium on Integrated network management IV table of contents*, pages 250–261, 1995.

[22] Jon Kleinberg. Detecting a network failure. In *Proc. 41st Annual IEEE Symposium on Foundations of Computer Science*, pages 231–239, 2000.

[23] R. R. Kompella, J. Yates, A. Greenberg, and A. C. Snoeren. Ip fault localization via risk modeling. *Proc.Networked Systems Design and Implementation*, 2005.

[24] James F. Kurose and Keith W. Ross. *Computer Networking Complete Package (3rd Edition)with study companion*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006.

[25] D. Larrabeiti, R. Romeral, I. Soto, M. Uruena, T. Cinkler, J. Szigeti, and J. Tapolcai. Multi-domain issues of resilience. *Transparent Optical Networks, 2005, Proceedings of 2005 7th International Conference*, 1, 2005.

[26] Earl Lawrence, George Michailidis, Vijayan N. Nair, and Bowei Xi. Network tomography: A review and recent developments. In *In Fan and Koul, editors, Frontiers in Statistics*, pages 345–364. College Press, 2006.

[27] Patrick Lincoln and Phillip Porras. Privacy-preserving sharing and correlation of security alerts. In *In USENIX Security Symposium*, pages 239–254, 2004.

[28] Yehuda Lindell, Rehovot Israel, and Benny Pinkas. Privacy preserving data mining. In *Journal of Cryptology*, pages 36–54. Springer-Verlag, 2000.

[29] Arvind Narayanan and Vitaly Shmatikov. Shmatikov how to break anonymity of the netflix prize dataset. arxiv cs/0610105, 2006.

[30] P. Reynolds, J. L. Wiener, J. C. Mogul, M. K. Aguilera, and A. Vahdat. Wap5: black-box performance debugging for wide-area systems. *Proceedings of the 15th international conference on World Wide Web*, pages 347–356, 2006.

[31] S. J. Russell and P. Norvig. *Artificial intelligence*. Prentice-Hall, 2003.

[32] C. E. Shannon. Communication theory of secrecy systems. *Bell System Technical Journal*, 28(4):656–715, 1949.

[33] N. Spring, D. Wetherall, and T. Anderson. Reverse engineering the internet. *ACM SIGCOMM Computer Communication Review*, 34(1):3–8, 2004.

[34] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring isp topologies with rocketfuel. In *In Proc. ACM SIGCOMM*, pages 133–145, 2002.

[35] M. Steinder and A. Sethi. Distributed fault localization in hierarchically routed networks. *Management Technologies for E-commerce and E-business Applications: 13th IFIP/IEEE International Workshop on Distributed Systems: Operations and Management, DSOM 2002, Montreal, Canada, October 21-23, 2002: Proceedings*, 2002.

[36] M. Steinder and A. S. Sethi. Probabilistic fault localization in communication systems using belief networks. *IEEE/ACM Transactions on Networking (TON)*, 12(5):809–822, 2004.

[37] M. Steinder and A. S. Sethi. A survey of fault localization techniques in computer networks. *Science of Computer Programming*, 53(2):165–194, 2004.

[38] M. Steinder and A. S. Sethi. Multidomain diagnosis of end-to-end service failures in hierarchically routed networks. *IEEE Transactions on Parallel and Distributed Systems*, 19(1):126–144, 2008.

[39] M. Steinder and AS Sethi. Increasing robustness of fault localization through analysis of lost, spurious, and positive symptoms. *INFOCOM 2002.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies.Proceedings.IEEE*, 1, 2002.

[40] M. Steinder and AS Sethi. Probabilistic fault diagnosis in communication systems through incremental hypothesis updating. *Computer Networks*, 45(4):537–562, 2004.

[41] Malgorzata Steinder and Adarshpal S. Sethi. The present and future of event correlation: A need for end-to-end service fault localization, 2001.

[42] L. Subramanian, S. Agarwal, J. Rexford, and RH Katz. Characterizing the internet hierarchy from multiple vantage points. *INFOCOM 2002.Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies.Proceedings.IEEE*, 2, 2002.

[43] L. Sweeney. k-anonymity: A model for protecting privacy. *INTERNATIONAL JOURNAL OF UNCERTAINTY FUZZINESS AND KNOWLEDGE BASED SYSTEMS*, 10(5):557–570, 2002.

[44] D. G. Thaler and C. V. Ravishankar. An architecture for inter-domain troubleshooting. *Journal of Network and Systems Management*, 12(2):155–189, 2004.

[45] Y. Vardi. Network tomography: Estimating source-destination traffic intensities from link data. *Journal of the American Statistical Association*, 91(433):365–377, 1996.

[46] H. Wang, Y. R. Yang, P. H. Liu, J. Wang, A. Gerber, and A. Greenberg. Reliability as an interdomain service. *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 229–240, 2007.

[47] Rebecca Wright and Zhiqiang Yang. Privacy-preserving bayesian network structure computation on distributed heterogeneous data. In *In Proc. of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 713–718. ACM Press, 2004.

[48] Ming Zhang, Chi Zhang, Vivek Pai, Larry Peterson, and Y Wang. Planetseer: Internet path failure monitoring and characterization in wide-area services. In *In OSDI*, pages 167–182, 2004.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Defense Information Systems Agency
   Arlington, Virginia

4. Army Research Lab
   Adelphi, Maryland

5. Space and Naval Warfare Systems Center
   San Diego, California